CHALMERS
UNIVERSITY OF TECHNOLOGY

Data-driven machine learning model for propeller design

Applying supervised regression algorithms to predict propeller blade geometry

Master's thesis in Mobility Engineering

Rasmus Andersson

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's Thesis in Mobility Engineering

Data-driven machine learning model for propeller design

Applying supervised regression algorithms to predict propeller blade geometry

Rasmus Andersson

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences
Division of Division Name
Chalmers University of Technology
Gothenburg, Sweden 2025

Data-driven machine learning model for propeller design
Applying supervised regression algorithms to predict propeller blade geometry
Rasmus Andersson

Supervisor: Tobias Huuva, Berg Propulsion AB
Examiner: Professor Rickard Bensow, Department of Mechanics and Maritime Sciences

Master's Thesis 2025
Department of Mechanics and Maritime Sciences
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2025

Data-driven machine learning model for propeller design
Applying supervised regression algorithms to predict propeller blade geometry
Rasmus Andersson

Department of Mechanics and Maritime Sciences
Division of Division Name
Chalmers University of Technology

Abstract

The high demand for innovative and high performance propellers for marine vessels motivates propeller designers to strive for a more efficient and less resource costly design process. To create a more robust process, propeller designers look towards potential resource optimization tools in the form of calculators, optimization algorithms and more. In recent years the topic of machine learning have made it's way into the propeller design field as a multifunctional tool that can be suited for many different tasks as long as the provided data exists. This thesis aimed to use supervised machine learning regression models to generate the first geometric blade profile that lay the starting basis for all propeller design processes.
The methodology included the creation of a data set based on high performing controllable pitch propellers, where geometric blade profiles were used to shape the in and output data for the models. Random forest, XGBoost, Neural network and support vector regression models were all used in the machine learning pipeline to make predictions on the blade geometry. Suitable performance metrics and cross-validation was plotted and used to ensure accuracy in the predictions while also providing interpretability to the models. The results from this study indicated that the highest performing model could explain 70% of the data-set variance. While not bad the results could be improved and it is believed that a lack of data quantity is the main factor holding the machine learning models back. More research and different approaches to the topic is needed to fully explore the possibilities of machine learning within the propeller designing field. This study provides valuable insight into the potential uses of machine learning and the specific performance of a set of models for smaller data-sets, resulting in a usable prediction tool that contribute to the propeller design process.

Keywords: Machine learning, Propeller design, Propeller geometry, regression methods

Preface

In this study, regression type machine learning models were tasked with predicting geometric propeller blade data. The study involved the following python machine learning libraries: Sklearn and XGBoost. The project was carried out in Berg Propulsion AB, Öckerö, Sweden between January and May 2025.

Acknowledgements

I am sincerely grateful to my supervisors and everyone who has extended their help and contributed to this master thesis. Your insightful suggestions and continued support throughout the project have helped me grow and has been critical for the result of the thesis. I am extra grateful to Berg Propulsion AB who gave me the opportunity to carry out my research on this interesting topic while also helping with valuable resources and tools. The project has given me great amount of knowledge and insight both into the subject, but also into previous course works that I have carried as a student of the Naval architecture master program at Chalmers.

It has been an incredible journey and i would like to extend my heartfelt thanks to the following people:

- Professor Rickard Bensow from the Department of Mechanics and Maritime Sciences, Chalmers University who has served the role as examination and have provided valuable academic support. I am deeply grateful for all the guidance and continuous assistance that has been provided on all matters related to the thesis, which has been proven to be crucial for the refinement and direction of the study.
- Tobias Huuva from Berg Propulsion AB who as served as project supervisor and provided expert insight and knowledge for the project. The constant support and curiosity into the research have made a deep impact on the shape of the study. I really can not thank enough for all the discussions and answered questions that have helped in moving the project forward but also helping me deepen my understanding on both the topic and propeller design as a whole.
- The core competence team at Berg Propulsion AB who has come to me with valuable suggestions and interesting discussions which has helped in moving the project forward in many ways. Their knowledge on the propeller design process has helped deeply for practical implementations in the project and for my understanding on a detailed level.

Finally I would like to thank my family and friends for their wonderful support throughout this fun but difficult project as a part of my Master's education.

Rasmus Andersson, Gothenburg, May 2025

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

ANN	Artificial Neural Networks
BET	Blade Element Theory
CFD	Computational Fluid Dynamics
CPP	Controllable Pitch Propeller
EAR	Expanded Area Ratio
EVS	Explained Variance Score
GHG	Greenhouse gas
IMO	International Maritime Organization
MAE	Mean Absolute Error
MLP	Multi-layer Perceptron
MSE	Mean Squared Error
NN	Neural Networks
RPM (rpm)	Revelations per minute
RSS	Residual Sum of Squares
SVM	Support vector machines
SVR	Support vector regression
TSS	Total Sum of Squares
XGBoost	Extreme gradient boosting

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Propeller design

A	Area
A_D	Developed area
A_E	Expanded area
A_p	Projected area
C_D	Drag coefficient
C_L	Lift coefficient
D	Diameter
J	Advance coefficient
K_Q	Torque coefficient
K_T	Thrust coefficient
P	Pitch
P/D	Pitch to diameter ratio
Q	Torque
R	Total radius
T	Thrust
V	Velocity
V_A	Average water inflow
W	Relative velocity
Z	Number of blades
c	Chord length
n	Revolutions per minute (rpm)
p	Pressure
r	Point on radius
r_h	Hub radius
ρ	Water density
θ	Pitch angle
θ_{nt}	Nose-tip pitch angle
θ_0	Effective pitch angle
β_i	Hydrodynamic pitch angle / angle of attack
$\dot{m}$	Mass flow rate
η_0	Open water efficiency

Machine learning

C	Regularization parameter
G	Gradient
H	Hessian
L	Loss function
M	Number of iterations
P	Number of neurons in input layer
R^2	Coefficient of determination
T	Number of trees
$V^{(h)}$	Hidden layer neuron output
$b^{(h)}$	Hidden layer scalar
e	Residual
$g^{(h)}$	Hidden layer activation function
n	Number of predictions
n_{leaf}	Number of leafs
ϵ	Mean-zero random error
ϵ_{marg}	Error margin
η	Learning rate
θ	Initial prediction
α, α^*	Lagrange multipliers
ξ, ξ^*	Slack variables
β	Coefficient
$\overline{x}, \overline{y}$	Sample means
$\hat{w}$	Weight
$\hat{y}$	Predicted value
y	Actual value

Contents

List of Figures

List of Tables

1

Introduction

The maritime industry is in a state of constant evolution, driven by the need for innovative solutions to meet growing demands for efficiency, sustainability, and performance. As new technologies and methods emerge at a rapid pace, one of the most critical areas of focus is the propulsion system. This system plays a key role in a ship's ability to move efficiently through water, maneuver effectively, and minimize environmental impact during operation (Bašić et al., 2024).

Central to the propulsion system is the propeller, a vital component that directly influences the overall performance of the vessel. With the industry's ongoing advancements, there is a rising need for propeller designs that deliver greater strength, improved efficiency, reduced noise, and lower weight, while meeting stricter environmental standards (Pan et al., 2023). However, designing these next-generation propellers is complex and multifaceted.

Propeller design currently depends on Computational Fluid Dynamics (CFD) simulations and iterative evaluations to optimize performance. Although this approach is effective, it is both time-consuming and computationally expensive, highlighting the need for more efficient and cost-effective design methods (Tadros et al., 2024a).

One tool that can be potentially implemented is machine learning (ML) models. An ML model which can learn from previous decisions in terms of propeller design and based on its training generates several iterations of propellers. This would optimize the design process by reducing the number of evaluations required to reach the final solution (Doijode et al., 2022). The ML model would continue to learn as more and newer data become available in the future, making it a tool with the ability to become more accurate in its prediction as time goes on.

The following sections will present and explain the background, aim and limitations of this thesis with the goal of providing insight into the subject's history, challenges, goals and why this subject is studied.

1.1 Background

Carlton (2018) describes the long history of marine propellers, how early designs,

theories and experiments from Archimedes and Leonardo da Vinci are still valuable for research in current years and to quote *"Although the propeller normally lies well submerged out of sight and therefore, to some extent, also out of mind, it is a deceptively complex component in both the hydrodynamic and structural sense"*.

In the current day propeller manufacturers and designers are shifting more and more towards shape adaptive propellers while also aiming for an improved hydroacoustic optimization. This means for the most part that more controllable pitch propellers (CPP) with reduced noise emissions are more sought after by ship owners. To adapt to newer design criteria, a clean-slate propeller geometry that fit within the margins must be designed.

This new geometry often does not have open water performance curves available since it cannot fall back on legacy designs. Starting from scratch is costly both financially and in the time it takes to create new designs, so there is always a need for improving and optimizing the design process (Doijode et al., 2022).

The collaborator for this thesis, *Berg Propulsion AB*, seeks constant improvements, and just like other propeller designers need new tools to handle the increasing demand for complex and modern propellers. An area in which it is believed that such tools can be found is in the subject of ML. A ML model contains the flexibility to handle complex and non-linear data while continuously being trained based on new designs (Tadros et al., 2024b). *Berg Propulsion AB* Intends for ML tools to take on the initial design stage of propeller development. This is done with the intent to reduce the time that it takes for the propeller designer to make an initial design that fits within rough criteria and instead let the ML model do it based on previous designs. This is believed to give more time for designers to optimize, refine and experiment with their designs, which both optimizes the process and allows for greater innovation and customization in the final propeller designs.

The possibility of implementing ML models to perform the work previously done by propeller designers is also an ethical question and not all positive. Potentially replacing human workforce due to automation technologies can be seen as unethical, but it is believed that the positives far outweigh the negatives of implementing such a tool. The reasoning being that propeller designers are highly valued for their ingenuity and problem solving skills, while the need for ML tools stems from need to reduce unnecessary work tasks.

1.2 Purpose

The purpose of this thesis is to compare the accuracy and performance metrics of ML models with the task of predicting key geometrical parameters for propeller blade shape. The data set used by the models is a product of raw geometric propeller data provided by *Berg Propulsion AB* that will be transformed into usable data suitable for ML.

The process of implementing ML models to predict geometrical data for propellers is applied with the goal of determining the usefulness of ML in a propeller design environment.

1.3 Scope

In this thesis the propellers used as a base for the data set are all shaft-line main controllable pitch propeller designs that have been confirmed and tested to meet Berg Propulsion performance criteria. This is to limit the variance and noise present in the data set. The conclusions then drawn from the results of this thesis are only applicable to Berg Propulsion Main shaft-line propellers or propellers with similar characteristics.

As stated the models will purely be tasked with predicting geometrical data for the blade shape of a propeller. This means that the goal is to predict the shape of the propeller and not the hydrodynamic performance of the propeller. There is also not any form of optimization of the propeller shape, the results are pure regression prediction of shape and not function. If predictions on the performance where to be included in the thesis, it is then believed that the scope would be to large and complex.

The data set is not pre-constructed for this thesis project and the creation and configuration of the data set will be a part of the methodology. This opens up for the involvement of feature engineering techniques and the data set it self being a variable instead of a constant. A consequence of this approach together with the unique characteristics of the provided propellers is that the methodology will be difficult to reproduce in other contexts.

The data provided by *Berg Propulsion AB* is as stated in a raw format describing only xyz-coordinates over radial sections, meaning that necessary simplifications will have to be done for the data to be usable. The simplifications enables the ML models to grasp relationships in the data easier. The simplifications add another layer of difficulty in terms of reproducing the results and open up for potential noise and error, without the simplifications it is believed that the amount of data required would have to be more than what was available. The end result of this approach is that the final predictions will ultimately be predictions based of simplifications and not the full geometrical profile of the propeller blade.

For ML models the processing power can serve as a limit on how complex configurations on models can be. For this thesis the processing power will remain constant throughout the methodology due to the computational options available, meaning that all results will based on the output of a single computer with the same hardware for all tests.

2

Theory

The following chapter will lay out all necessary theory required to get an understanding for ML in a propeller designing environment as well as project specific information which all lay the base for this thesis.

This chapter will be divided into propeller design and ML specific information with corresponding sub-sections related to each topic containing theories, formulas and figures while keeping the information categorized.

2.1 Propeller design

In 2023 the International Maritime Organization (IMO) published a strategy on how to tackle greenhouse gas emissions (GHG) in the shipping industry. A strategy meant to make shipping more sustainable and environmentally friendly by encouraging shipowners to promote development and research related to lowering the GHG emissions from ships. One of the larger goals that IMO mentioned in the declared strategy is to reduce the carbon intensity on average of international shipping with at least 40% by 2030. This is done in correlation to reach the long standing temperature goal set out in article 2 of the Paris agreement (IMO, 2023). To lower the carbon intensity takes effort from shipbuilders and shipowners, new technologies must be researched and implemented. Goals like the one presented by IMO is what moves the industry forward, that of course also means the development of new propulsion systems and propellers. Today engineers look on propeller in a mechanical and hydrodynamical sense to optimize efficiency and delivered thrust utilizing theories and methods presented in the sections below.

2.1.1 Geometrical parameters

Carlton (2018) on the importance of propeller geometry states, *"To fully appreciate propeller hydrodynamic action from either an empirical or theoretical standpoint, it is essential to have a detailed understanding of propeller geometry and the corresponding definitions used"*.

To begin the development of a propeller a certain set of operational parameters must be acquired as a starting point for the process. These parameters lay the foundation for what the purpose of the propeller will be as well as providing enough information for calculating the first iteration of a blade shape for the propeller. Khose (2020) divides the factors to consider in the starting phases of propeller design into two aspects, geometrical and miscellaneous parameters. Geometrical parameters which define the blade shape and lay the basis to its performance in the water. Parameters include,

- Propeller diameter
- Blade surface area
- Blade outline
- Pitch & pitch/diameter ratio
- Skew & rake angle
- Camber

Whereas miscellaneous parameters are not as defined and can differ from project to project, but in general the miscellaneous parameters are ship and class related. Examples for some miscellaneous parameters are ice class, ship type and shaft rotations per minute (RPM). These parameters hold significant importance for the propeller development in terms of rule set and accepted margins.

RPM & torque

RPM and torque in a geometrical sense are input parameters that stem from ship machinery and propulsion systems. The RPM of the shaft does not have a connection to the design speed of the ship, meaning how many knots the vessel reaches during operation. Torque is also connected to the machinery system onboard and is a reference to how much rotational force the propulsion system can output. This is important for propeller design since the machinery must be able to rotate the propeller in the water for the ship to operate, meaning that it is a balance between the output torque and how hard it is to turn the propeller in the water.

The changes in RPM for a propeller can drastically change how it behaves in the water, the force generated, and the efficiency of the propeller. Research shows that an increase in RPM can provide a higher mass flow rate and thrust compared to lower RPM, meaning that the amount of effective force that propels the vessel forward is

increased. A higher RPM can provide higher usable forces, but come with the side effect of a higher potential cavitation area which can damage the propeller over time (Moon et al., 2014).

All parameters in propeller design is in some way connected, meaning that changes in one parameter can lead to changes in another. RPM and torque are no different and their respective values have great impact on the geometrical profile as well as hydrodynamical performance of the propeller. Due to the connection to machinery system onboard and the ships capabilities RPM and torque are to an extend predetermined before propeller development. While possible to change with changes to propulsion system, RPM and torque are good input parameters for the first design iteration and can be seen as a general outline for other geometrical parameters to follow (Khose, 2020).

Pitch

Pitch is another essential parameter for the propeller design and characteristics. Pitch can be divide into pitch and pitch angle which represents two different things. The pitch is a measurement of the distance which a point on the propeller blade travels during a rotation, while the pitch angle is the main factor for the achieved distance. Fig 2.1 visualizes how the propeller blade turns in the shape of a helix along a cylinder, the distance created is the pitch (P) while θ is the pitch angle. With this it is observable how θ can increase the distance between points along the helical surface and thus increasing P. With a higher θ value then the distance that the propeller travels in the water increases and vise versa for a lower pitch angle (Carlton, 2018).

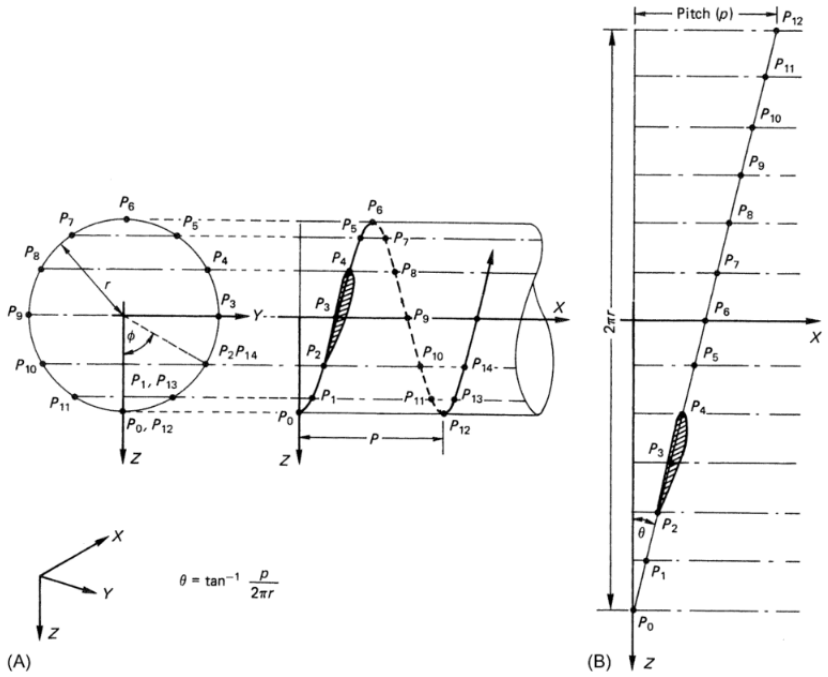


Figure 2.1: (A): A helix representation of pitch on a cylinder of radius r. (B): The development of helix on the cylinder plotted on a coordinate field (Carlton, 2018).

There are several pitch angle definitions that are important in propeller analysis such as, nose-tip pitch, effective or "no-lift" pitch and hydrodynamic pitch. The different pitch angles are shown in 2.2, where nose-tip pitch line is the most commonly used reference line today by propeller designers since it defines blade sections and its angle is noted as θ_{nt}. The face lift line shown in 2.2 is a pitch angle that is rarely used today with it being a basic tangent to the blade section's pressure side surface and holds no hydrodynamic significance. The effective pitch line of the blade section corresponds to the conventional aerodynamic no-lift line, this line is where if the incident water flowed along it no lift would occur from the aerofoil section. The effective pitch angle (θ_0) is the summation between the nose-tail pitch angle and the three-dimensional zero lift angle of the blade section. The final angle is the hydrodynamic pitch angle (β_i) which is a hydrodynamic inflow and not a geometric property where the angle is defined as where the incident flow encounters the blades section. Under normal circumstance effective and hydrodynamic pitch angles are not shown on propeller drawings (Carlton, 2018).

The correlation between the different pitch definitions mentioned can be written in a simplified way as,

Effective pitch angle = Nose-tip pitch angle + 3D zero lift angle = Hydrodynamic pitch angle + 3D zero lift angle + angle of attack of section

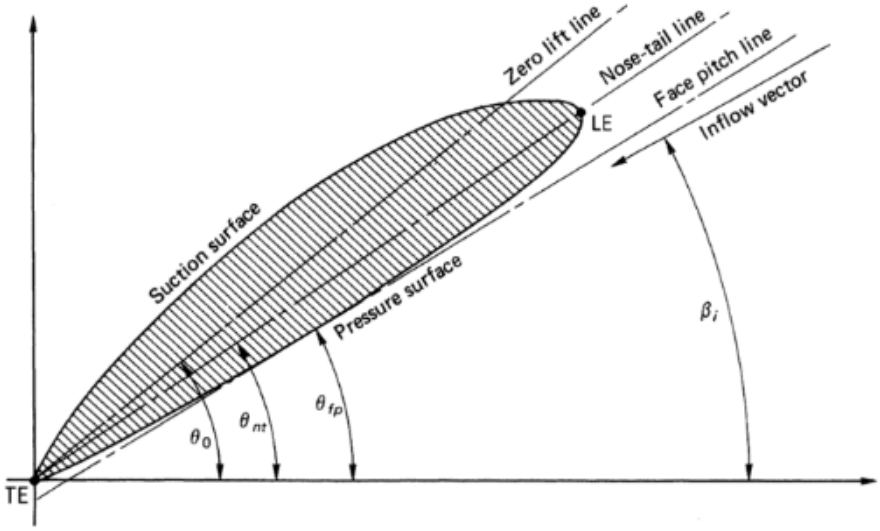


Figure 2.2: Pitch lines on propeller blade (Carlton, 2018)

The diameter of the propeller is often defined before the propeller design stage starts and is defined by the wharf building the vessel. The diameter is important in its relation to other parameters when building the propeller geometry. The different curves and lines that make up the geometry are often defined into correlation with the diameter or radius or the propeller. For many propeller drawings and calculation methods a dimensionless radial coordinate x is used for points and curves,

$$x = \frac{r}{R} \qquad x \in [0, 1], \tag{2.1}$$

where R is the total radius of the propeller blade and r is a specific point on the blade. If a point on the propeller is located on the propeller hub it is defined as r_h(Birk, 2019).

By dividing pitch with the diameter of the blade, the commonly defined P/D ratio is obtained. The P/D ratio is used to specify the pitch for the propeller and normally the ratio falls within values of 0.5 to 1.4 with exceptions for very high speed vessels. With a higher P/D value, meaning that the length of the pitch is considerably larger than the diameter of the propeller the onboard machinery will need to be able to provide the sufficient torque to rotate the propeller. Finding the right P/D ratio is a efficiency problem and defining a badly fitting P/D ratio for the propeller and ship configuration can lead to poor performance for the ship. P/D ratio is therefore finely tuned and optimized during propeller development (Birk, 2019).

Blade area and Blade outline

What could be considered the true geometry of the blade defined in drawings in different ways is the blade area and blade outline. The outline and area of the propeller blade is another way of denoting the width of the blade from different perspectives

and the width of the blade is developed in close reference to the cavitation criteria of the propeller. The propeller blade is designed in sections to be able to properly study the sections as well as meeting criteria such as cavitation, and at the end of the process the section needs to be fitted together which makes up the propeller outline and in turn the area of the blade (Carlton, 2018). As stated before, the propeller outline that together describes the blade shape can be defined in different ways, with the four basic outlines being:

- The projected outline,
- The developed outline,
- The expanded outline,
- The swept outline.

The projected outline as seen in fig 2.3 is the outline viewed in a two dimensional sense when observed along the shaft. It is in simplicity a view of the blade from an y-z standpoint without taking depth created by the pitch into consideration. The developed outline drawn in a dot line in fig 2.3 is a flattened viewed of the propeller blade. This means that the depth of the propeller is taken into consideration but it is flattened or in other words the pitch is assumed to be equal to zero to create a two dimensional drawing. The expanded outline is in reality not a true outline but instead a plotting of the chord lengths at their correct radial stations about the directrix. The helical nature of the propeller blade is not represented in this view meaning that the pitch i assumed to be zero. The Expanded outline is a useful tool to represent the blade sections as they are often plotted against the chord lengths of the propeller. Finally the swept outline is used to represent stern frame clearances and is especially important for heavily skewed propellers due to if not controlled properly the skew-induced rake can lead to a significant overhang which is undesirable (Carlton, 2018).

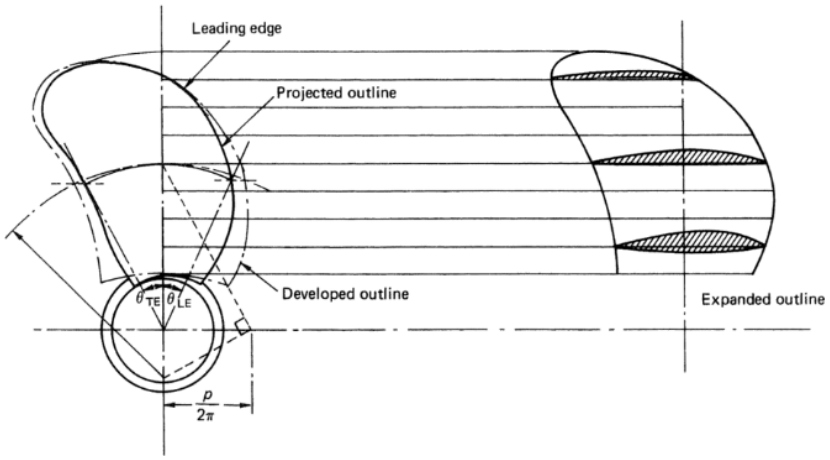


Figure 2.3: Blade outlines (Carlton, 2018)

The projected, developed and expanded outline have corresponding areas over the

blade which starting with the projected outline which is calculated as:

$$A_p = Z \int_{r_h}^{R} (\theta_{TE} - \theta_{LE}) r \; dr. \tag{2.2}$$

Where Z is the number of blades and the difference between trailing edge pitch angle and leading edge pitch angle represents the helical nature of the propeller blade while the integral between the propeller hub and tip represents solely the blade area. This equation is not very relevant today but has historically been used to optimize the propeller blade area for cavitation through thrust loading per unit projected area. The developed and expanded area are almost equal and in many cases it is practical to assume that the two are equal,

$$A_D \approx A_E. \tag{2.3}$$

Research have been done in regards to producing empirical relationships for the developed area. A relationship for developed area, presented by Burril is defined as:

$$A_D = \frac{A_P}{(1.067 - 0.229 \frac{P}{D})}. \tag{2.4}$$

The equation that is produced for non skewed forms is made through a fraction between the projected area and constants which include the pitch to diameter ratio. In general the developed area is larger than the projected area. The expanded area is the simplest due to it normally being quoted and is represented by the relationship as follows,

$$A_E = Z \int_{r_h}^{R} c \, dr, \tag{2.5}$$

where c is the the chord length for the coordinate. Normally using Simpson's rule procedure which defines the blade in 11 sections is sufficient. The areas are important for the calculation of the propeller geometry, but the ratio between the areas defined above and the propeller disc area is critical as input parameters and are often set before initial calculations and later through iterations is altered to yield better results. The most important ratio that is used in almost all cases is the expanded area ratio (EAR) which is an indicator for how much area of the propeller disc that the propeller takes up (Carlton, 2018), and is defined as:

$$EAR = \frac{A_E}{A_0} = \frac{4A_D}{\pi D^2}. \tag{2.6}$$

Where EAR is as said a representation of how much of the circular propeller disc that the blade occupies, meaning that a higher EAR is equal to a larger blade in terms of blade area. The changes in EAR are then directly tied to the mass of the propeller which can be a limiting factor in terms of structural integrity. A higher EAR also has a direct connection the damping coefficient of the propeller as fig 2.4 showcases. The damping coefficient represents how well the propeller dissipates energy, so a higher damping coefficient can be seen as a reduction in efficiency but comes with an increase in stability, noise reduction and less vibrations. Whereas

a lower damping coefficient leads to a higher propeller efficiency but the increased vibration and instability can lead to more wear and tear in comparison (Abbasi et al., 2024).

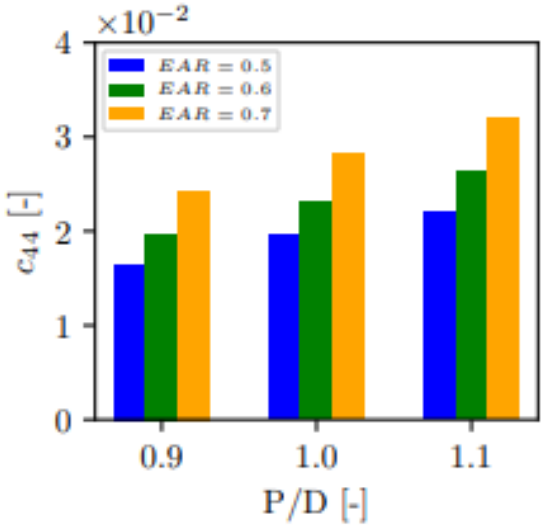


Figure 2.4: Effect of EAR and P/D on damping coefficient (Abbasi et al., 2024)

Skew & rake

Skew and rake are angles describing how the blade is positioned in regards to the propeller hub which gives the propeller a helical and curved geometry. The rake angle is as seen in fig 2.5 the angle of the line that is running from the center of the hub to the tip of the blade, also called the generator line. If the rake angle is equal to zero then the generator line is perfectly perpendicular to the propeller hub. When the rake is angled towards the aft of the vessel then it is a called positive rake and if angled towards the fore then it is a negative rake.

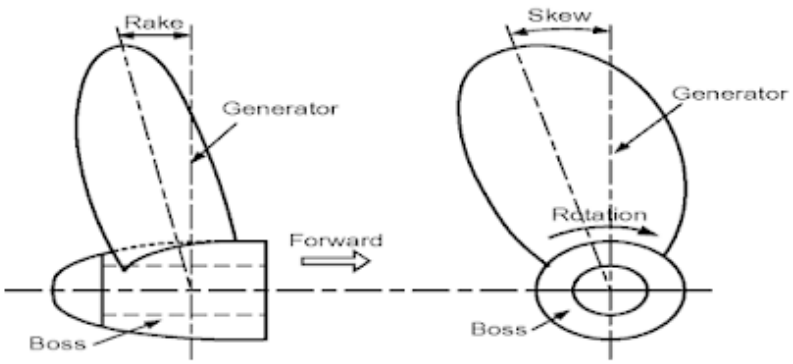


Figure 2.5: Skew and rake angles on a propeller blade (Sadiq et al., 2025)

Skew follows the same concept as rake but from the viewpoint that can be seen in the propeller located to the right in fig 2.5. As the generator line gets an increased

angle the blade obtains a curved shape with the tip not being perpendicular to the hub. (Sadiq et al., 2025).

Different propellers have different optimal skew, which will have to be developed through testing and CFD calculations where all parameters are involved. For example a highly skewed propeller have been confirmed to operate more efficiently at lighter loads than at heavier loads signifying the importance of fitting the skew after the operating conditions (Boumediene et al., 2018). A starting skew angle is often provided as an input before propeller development which serves as a starting point to the iterative process to finding the perfect fit.

Just as with the skew, the rake angle needs to be optimized depending on operating parameters. What sets rake apart from skew in that sense, is that the rake has a more significant interaction with the hull of the ship. A higher rake angle will give the propeller more depth which in turn will alter the amount of 3-dimensional space that the propeller occupies, therefore rake needs to be carefully considered when it comes to hull interaction and propeller placement. Research shows that an increase in rake can lead to a higher efficiency and thrust, but a too large rake angle can lead to increased noise levels. For a DTMB 4119 marine propeller the optimal rake angle is considered to be between approximately 8-12 degrees which serves as an example for how small the margins are for an optimized rake (Ebrahimi et al., 2019).

Camber

The final geometric parameter camber essentially describes how the blade is bending and is a critical parameter for determining the amount of lift the propeller produces at certain angle of attacks. Carlton (2018) clarifies camber and states how a propeller with zero camber is fully flat where as a positive camber gives areas a convex nature and negative camber a concave nature. The camber is defined across radial sections through the camber line, where the camber line is drawn on the blade from the leading edge to the trailing edge. The line is also drawn in the middle of the blade so that the thickness from camber line to either suction or pressure side is equal. Camber is almost always positive but varied across the radial sections to give the propeller a unique profile that is hydrodynamically optimized. In addition camber is a parameter that is highly tied to the propellers operating condition and can vary heavily depending on the type of ship or its purpose during operation.

2.1.2 Hydrodynamic principles

While the geometric parameters describe the shape and form of the propeller blade, the hydrodynamic principles decide its performance. The two are directly dependent on each other, changes in the geometry will have big effects on the propellers hydrodynamic performance regarding efficiency, thrust, torque, and cavitation to name a few. The relationship between the two is what pushes propeller development forward and it is essential to have knowledge of the hydrodynamic principles, what they are, how are they controlled, and what causes changes in them. Hydrodynamics is a complex and deep field and for the case of this thesis main focus lying

in other fields hydrodynamics will not be explained in a deeply detailed level. This section will showcase the principles effect, how they are controlled and provide an understanding as for how they relate to the propeller geometry.

Thrust

The main force generated by the propeller which pushes the ship forward is thrust. It is a force resulting from the pressure difference between inflow and outflow sides that occurs as the propeller rotates, while also taking the disc area of the propeller into consideration.

$$T = A_0 \cdot \Delta p \tag{2.7}$$

With eq 2.7 in mind then thrust is directly tied to the disc area of the propeller, meaning that a larger diameter will result in a higher thrust. As for the pressure difference that occurs, it is the result of the difference in water velocity flowing into the propeller and the velocity of the water flowing out of the propeller. The way that the water reacts to the propeller can be described in three stages, where the first stage is the inlet. In the water inlet the velocity of the water (V_1) is equal to the average inflow or V_A. The speed of the water changes as it reaches the second stage, the propeller. As the flow reaches the point right in front of the propeller it gains in velocity due to the suctional forces of the propeller which moves the water in its direction, increasing its velocity. This second stage velocity can be explained as $V_2 = V_A(1 + a)$ where a is the added velocity produced by the geometry. The third stage is further downstream of the propeller after the water has been pushed out and the velocity here is described as $V_3 = V_A(1+b)$ where b is the gained velocity from the water being pushed out of the propeller. Generally b is larger than a due to the mechanical induced forces that comes from the rotating propeller as well as the pitch. With this in mind, momentum theory can be applied by firstly utilizing newtons second law,

$$T = \dot{m}(V_3 - V_1), \tag{2.8}$$

where $\dot{m}$ is the mass flow rate through the propeller disc. $\dot{m}$ is the product of water density,Area and velocity which results in the following relationship,

$$T = \rho \cdot A \cdot V_2(V_3 - V_1). \tag{2.9}$$

The application of Bernoulli's equation gives us the following equation,

$$p_\infty + \frac{1}{2} \cdot \rho \cdot V_1^2 = p_1 + \frac{1}{2}\rho \cdot V_2^2, \tag{2.10}$$

and for the downstream region,

$$p_\infty + \frac{1}{2} \cdot \rho \cdot V_3^2 = p_1 + \frac{1}{2}\rho \cdot V_2^2, \tag{2.11}$$

where p_∞ is an indication for very far upstream or downstream depending on equation. Subtracting the two equations for the upstream and downstream region results in the pressure difference,

$$p_2 - p_1 = \frac{1}{2}\rho(V_3^2 - V_1^2), \tag{2.12}$$

where p_1 is the pressure right in front of the propeller disc and p_2 is the pressure right behind it. These equations, which is the result of momentum theory, showcases that thrust is the result of changes in water flow velocity that the propeller creates, the water properties in terms of density and average inflow and the area of the propeller disc (Birk, 2019).

Torque

Torque can be described as a mechanical moment stemming from the machinery of the ship and the speed of which it rotates the propeller shaft. However, torque in a hydrodynamic sense is the moment that is needed to keep the propeller spinning and can be seen as a limit for just how much power the machinery must be able to produce to be able to rotate the propeller and shaft. If the power is not enough then the propeller will not move. The two definitions can be described as, mechanical torque which is the moment produced by the machinery system and hydrodynamic torque, often denoted as Q is how much mechanical torque is needed. The hydrodynamic torque parameter is a sum of resisting moments coming from the water and the propeller geometry. For example a very small diameter propeller with little rake requires much less torque compared to a larger diameter propeller with a higher rake angle. Torque as a limiting factor like this is often calculated through computational means such as CFD, which can take water characteristics, different hydrodynamic interactions as well as different states and operational parameters into consideration during its calculations (Faltinsen, 2005).

To explain torque in a mathematical sense, blade element momentum theory (BET) can be used. The application of BET in the case of propeller torque means to sum the lift and drag coefficients, C_L and C_D over the geometry of the blade,

$$Q = \int_0^R 0.5 \cdot \rho \cdot W^2 Z \cdot c(C_L \cdot \sin\beta_i + C_D \cdot \cos\beta_i) \cdot r \, dr, \tag{2.13}$$

where W is the relative velocity, Z stands for the number of propeller blades, the chord length is denoted as c and the relative velocity angle at domain inlet as β_i.

The lift and drag coefficients are local properties of each small blade section, meaning that they change across the sections of the propeller and can therefore be heavily influenced by the localized propeller geometry such as skew and section width. β_i from eq 2.13 is the angle of which the water hits the propeller blade which is a large factor affecting lift and drag. The angle between β_i and the pitch of the propeller is called the propellers angle of attack. The angle of attack is fined tuned during the propeller development to get the most fitting lift and drag for the propeller and its purpose (Benini, 2004)

Efficiency

The main result that is perhaps the most important to all propeller design is efficiency and more specifically open water efficiency which describes the propellers

performance during steady flow. Mathematically the open water efficiency is defined as,

$$\eta_0 = \frac{T \cdot V_A}{2\pi \cdot n \cdot Q} = \frac{K_T \cdot \rho \cdot n^2 \cdot D^4}{K_Q \cdot \rho \cdot n^2 \cdot D^5} \cdot \frac{V_A}{2\pi \cdot n} = \frac{K_T}{K_Q} \cdot \frac{J}{2\pi}, \tag{2.14}$$

where the relation ship between thrust, torque and average inflow is what defines the efficiency. K_T and K_Q are thrust and torque in a non-dimensional form while J is the advance coefficient, n is the propeller rpm and D the propeller diameter. T and Q are transformed into their non-dimensional forms through division of the propeller RPM, water density and propeller diameter as followed:

$$K_T = \frac{T}{\rho \cdot n^2 \cdot D^4} \tag{2.15}$$

$$K_Q = \frac{Q}{\rho \cdot n^2 \cdot D^5}. \tag{2.16}$$

Whereas the advance coefficient is a function of the average water inflow, RPM and diameter,

$$J = \frac{V_A}{n \cdot D}. \tag{2.17}$$

In propeller design it is very common to plot K_T, K_Q and η_o as a function of J, providing a visualization as how the different parameters change as the advance coefficient increases. These plotted curves are called open water diagrams and are unique to every propeller and serves as an important tool of understanding during propeller development how changes in parameters affect each other and what the operating point of the ship should be. Illustrative diagrams are important when it comes to providing and visualizing patterns, but in modern times computational calculations are used almost exclusively to optimize the parameters instead of visualizing them through the help of datasets of coefficients. This makes it possible to instead represent the curves as polynomials,

$$K_T = \sum C_T \cdot J^s \cdot (\frac{P}{D})^t \cdot (\frac{A_E}{A_0})^u \cdot Z^v, \tag{2.18}$$

where C_T, S, u and v are coefficients for a specific propeller series, an example being the standard Wageningen propeller series (Bertram, 2012).

Cavitation

The hydrodynamic phenomenon called cavitation is present in almost all forms of engineering containing liquids and it is especially important for propeller development where it acts as a limiting factor for the shape and performance of the propeller. Cavitation is the appearance of vapor cavities inside an initially homogeneous liquid medium, which occurs during very low pressure conditions. The concept of cavitation can be explained in theory in three steps. The first being a breakdown or void

creation, the second is the created void being filled with vapor and the third step is the eventual saturation with vapor. In reality distinguishing these three steps is very difficult as they happen almost instantaneously, but in simpler terms cavitation can be seen as a small implosion that occurs on different parts of the blade which contributes to the erosion of the blade (Franc & Michel, 2006).

There are different forms of hydrodynamic cavitation that occurs under different circumstances which are important to consider during propeller development. The different cavitation forms regarding the propeller are, bubble cavitation, sheet cavitation, vortex cavitation, and cloud cavitation. Franc & Michel (2006) describes how bubble cavitation is very localized on the propeller surface and that it happens as a vapor bubble, which is created in low pressure zones on the propeller reaches higher pressure areas and implode with massive force. Radosavljevic (2011) states that unlike bubble cavitation, sheet cavitation is not localized and can be observed as a form of vapor sheet that covers an area on the suction side of the propeller stemming from the leading edge of the blade. This form of cavitation is not as intense as other forms of cavitation but is more consistent due to geometry of the blade which creates the low pressure zone that is required for sheet cavitation to occur. Cloud cavitation is what happens when the normally stable sheet cavitation enters into an unstable state. This can happen due to changes in the current or flow as well as changes in operating parameters as the propeller is rotating. When the pressure becomes unstable or changes the vapor sheet enters into a chaotic state and rapidly and violently implodes. This form of cavitation is the most intense in comparison to the rest. Finally, vortex cavitation is a spiral of vapor present in the vortex core and can be very clearly observed in a cavitation tunnel as it forms long vapor threads from the tip of the propeller blades. This type of cavitation is not very intense in the sense of erosion and mainly affects the tip of the blade, but it can be a big contributor to the overall noise that comes from the operation of the propeller (Carlton, 2018).

Performance Curves

The hydrodynamic performance of the propeller is often visualized through different curves displaying how different parameters interact and how they affect the propeller. The performance curves all visualizes different aspects and parameters that are important during propeller development, often merging the line between geometric and hydrodynamic parameters. Below some of the more commonly used curves will be described, but there exists a lot more that touches more niche areas of propeller design.

Starting off with the open water curve which is based on eq 2.14 and displays thrust, torque and open water efficiency with regards to the advance coefficient. This plot creates a visualization for how the parameters are affected as the advance coefficient increases which is often used to find the best balance of the parameters with regards to performance. Efficiency is plotted against the pitch and blade area ratio to get a reference for the effect of changes in propeller geometry. The Bollard Pull curve explains the thrust and RPM with no advance coefficient present, which is important

for vessels that perform pulling operations or dynamic positioning. Cavitation curves are used to identifying under what operating conditions cavitation occurs. This is done by utilizing the cavitation number and plotting it against advance coefficient or thrust. There exists a lot of performance curves and test curves that are important during the iterative development of the propeller. The curves all aim to gather how the relation between parameters affect each other. While also providing insight into where one parameter can be more prioritized than another depending on the operating conditions and goal for the propeller.

2.2 Machine learning

The concept of ML is, as the name suggest, about computers learning from experience. In this context experience refers to pattern recognition and analysis ranging from identifying numerical trends to visual features in images and much more. ML and the process it will go through to read patterns and make predictions is in reality an extension of mathematical statistics utilizing computational methods. Naqa & Murphy (2015) describes ML as a process that utilizes input data to achieve a task without being literally programmed to produce a particular outcome. That the ML algorithm will continuously adapt through repetition with the goal of improving its architecture to better achieve the task it was set out to do. In this section the different relevant ML algorithms will be explained as well as how their respective process work and what tasks they specializes in with a focus on supervised learning.

2.2.1 Machine learning basics

As stated ML is in simplified terms a form of pattern recognition where the algorithm will iterate statistical calculations over a dataset to try and accurately predict an outcome. The process of iterating these calculations is to grasp the pattern of the data set is called training, and the generated prediction is called testing (Zhou, 2021). An important factor that regards learning and training is that the algorithm can only work with the data provided, meaning that if the data input consists of non relevant data then the testing results will not achieve the task since the model was tested on irrelevant data. This highlights the importance of a good and relevant dataset to the specified task. Assuming that the dataset used is relevant for the task then Naqa & Murphy (2015) describes how ML can in general be divided into multiple categories, supervised- and unsupervised algorithms are two among them and the most commonly used methods.

To explain the concept of supervised and unsupervised algorithms a simple sorting example with pictures of cats and dogs. In an supervised algorithm all pictures of the cats and dogs have a corresponding label stating that the pictures are indeed either a cat or a dog. During training the algorithm can recognize when it sorted a cat as a dog and vise versa and improve it's architecture to produce better results during testing. As for unsupervised, then the pictures will not have a label and the algorithm will sort only based on common factors between the pictures during training which can have undesired results during testing such as sorting the pictures

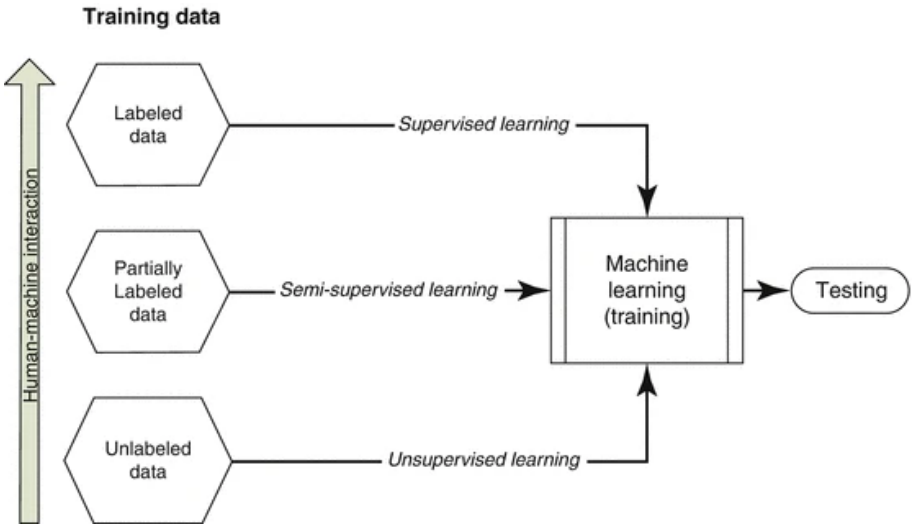


Figure 2.6: Description of ML structure with regards to human input (Naqa & Murphy, 2015)

by color. While the outcome in unsupervised algorithms may require more training to get the wanted results the supervised scenario requires more human input but a better result for the sorting task. The relationship between human input and respective model can be described by fig 2.6 where the concept of semi-supervised learning is present, which means that only some of the data is labeled.

ML can be trained on many different dataset to solve many different problems. All of these tasks fall under the subject of statistics and an ability to make predictions. They are also divided into subcategories which serves as a description for what type of problem it is. According to Wang (2016) the different subcategories of problems within supervised learning are classification and regression which are among the most common problems. For other problems within other domains like unsupervised learning even more subcategories for problems can be found.

Classification tasks are much like the dog and cat example where the algorithm aims to sort out data into different classes such as dogs or cats. The principle behind classification in slightly deeper terms can be explained through fig 2.7 where the data has been divided into orange and blue dots for clarity. The example showcases how a neural network handles a classification task by creating the white lines between orange and blue dots. Wang (2016) states that the white line present in the example is called the decision boundary which aims to as accurately as possible separate the different color dots.

In the example in fig 2.7 the classification problem is binary, meaning that there are only two possible categories to sort the dots into. When the task is more complex and can no longer be contained within the simple binary restraints it is called Multi-class classification. This adds more multi discrete outputs to the problem, an example being image classification with more than 1000 sorting categories (Mitchell, 1997).

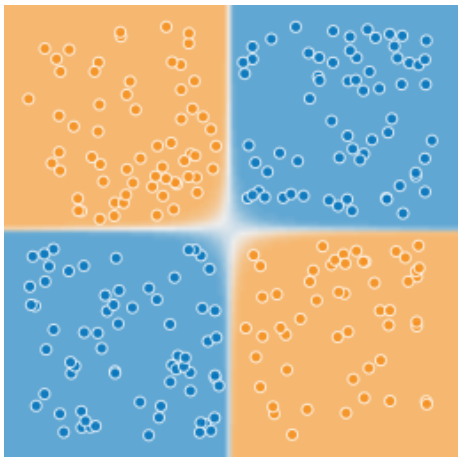


Figure 2.7: ML classification plot (Smilkov et al., 2025)

While the classification approach is great for sorting and dividing provided data into classes it is not ideal when it comes to predictions involving continuous or real-valued outputs. This leads into the second type of task under supervised learning, regression tasks. Rong & Bao-wen (2018) describes how regression tasks are solved by an algorithm predicting the continuous output with independent variable or variables as input. Essentially the algorithm is predicting where the next data point is located. This can be tied to a multitude of problems from predicting new points in curves to estimating the next days weather. The most common and well known example is linear regression. Riccardo & Flavio (2021) States that the concept of simple linear regression can be explained through,

$$y = \beta_0 + \beta_1 x + \epsilon, \tag{2.19}$$

where it is assumed that there is an approximately linear relationship between y and x. β are unknown coefficients and ϵ is the introduced mean-zero random error term.

The simple linear regression is iterated throughout all data creating coordinates of predictions on a coordinate field. Example of this can be predicting the sales of a product through the linear relationship of the products advertisement, resulting in a coordinate field for the amount of sales compared to advertisement. Eq 2.19 highlights how ϵ is present, which means that the predictions will not be fully accurate, but before the error is calculated the prediction itself must be done. In ML prediction there will in almost in all cases be an amount of error present, and to both improve and track the performance of the algorithm is of great importance. To make a prediction from the data points collected the coefficients β from eq 2.19 must be defined. This is done through fitting the linear model through the known data, essentially drawing a straight line through the data cluster which is a close as possible to the most data points. Riccardo & Flavio (2021) visualizes the concept through fig 2.8 which is an example of the relationship between TV advertisements and sales and highlights the fitted blue line which is calculated to be as close as possible to the most data points while keeping its linearity. There are many ways to define β_0 and β_1, but the least squares method is the most common one. Nwanganga

(2020) describes that if a smaller line is drawn from each data point plotted to the prediction line there will be an amount of difference in the y value. This is visualized by the smaller grey lines in fig 2.8 and the difference between the actual value y_i and the predicted value at the same x coordinate $\hat{y}_i$ is called the residual (e_i)

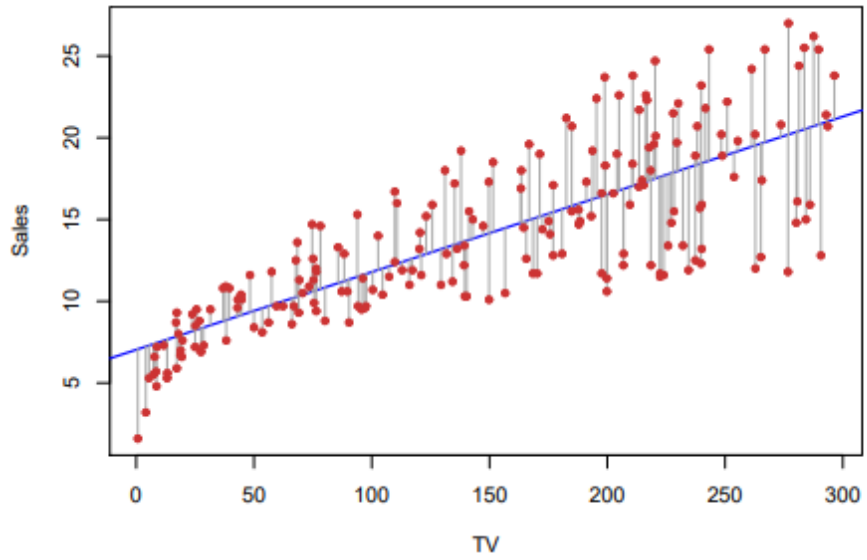


Figure 2.8: Linear regression fit example, utilizing least squares method.

The residual is another term for error is utilized in the least square method by firstly adding together the residual sum of squares (RSS),

$$RSS = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - \hat{\beta}_0 + \hat{\beta}_1 x_i)^2, \tag{2.20}$$

2here $\hat{\beta}$ are the coefficients for the prediction line. The optimal values for $\hat{\beta}_0$ and $\hat{\beta}_1$ is found where RSS is as small as possible. Nwanganga (2020) displays that the value of $\hat{\beta}_1$ that minimizes RSS can be calculated as,

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(x_i - \overline{x}_i)(y_i - \overline{y}_i)}{\sum_{i=1}^{n}(x_i - \overline{x}_i)^2} = \frac{\mathrm{Cov}(x, y)}{\mathrm{Var}(x)}, \tag{2.21}$$

where $\overline{x_i}$ and $\overline{y_i}$ is the sample means of x and y. Through solving for $\hat{\beta}_0$ in eq 2.19, both coefficients are optimized for the least amount of error. In many ML problems just one independent variable is not enough and many variables are used to make the final prediction. This scenario is called multiple linear regression and operates on the same principle as simple linear regression, but with som modifications to take more variables into consideration. The prediction for multiple linear regression is calculated as,

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \tag{2.22}$$

Riccardo & Flavio (2021) highlights how RSS is calculated in the same manner as in eq 2.20, but adding together all the values of eq 2.22 instead. It is also stated that

solving for the different values of β is better solved through matrix calculations,

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1n} \\ 1 & x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}, \tag{2.23}$$

The matrices made with regards to any number of variables, and are then used to optimize the values of β to minimize RSS as followed:

$$\hat{\beta} = (\mathbf{X^T X})^{-1} \mathbf{X^T y}. \tag{2.24}$$

Multiple linear regression can only plot two variables at once due to the limit of the three dimensional plane. If there is a need to visualize the different variables it will be through cross section cuts, meaning that all but two x variables are assumed constant while only observing the two variables. This reduces the amount of dimensions, easing illustration.

When a minimized value of RSS is obtained for the regression, the statistic R^2 can be calculated as in eq 2.25. Little (2019) states how the total sum of squares (TSS) is derived from the same squares as RSS but not cubed, which can be defined as $\sum(y_i - \overline{y}_i)$. The final result of R^2 is a value between 0 and 1 and serves as an indicator for how accurate the prediction is, where 1 is fully accurate and 0 the opposite. This form of error calculation of predictions serves as a base for determining accuracy in many ML models.

$$R^2 = 1 - \frac{RSS}{TSS} \tag{2.25}$$

It is also stated that linear regression methods is a foundation for many ML models handling regression tasks, but its simple and linear structure makes it incapable to handle situations where non-linearity is required. Therefore, many algorithms have been developed to better suit the complexity of more advanced problems that are unsolvable with the linear regression method.

Performance metric such as R^2 are important for interpreting why the predictions from a ML algorithm is what it is as well as to how accurate it is from multiple point of views. Chauvin & Rumelhart (2013) states that R^2, mean squared error (MSE), mean absolute error (MAE) and explained variance score (EVS) are among the most common performance metrics to be used. MSE is very similar to RSS described by eq 2.20 with the difference that the sum is divided by the amount of iterations. MAE is similar, but instead of the difference between y_i and $\hat{y}_i$ it is the absolute value between the two. EVS is represented as,

$$EVS = 1 - \frac{\mathrm{Var}(y_i - \hat{y}_i)}{\mathrm{Var}(y)}, \tag{2.26}$$

which is the variance of the difference between actual and predicted value divided by the variance of the actual value, resulting in a metric that explains how well a model understands total variance in the data.

2.2.2 Model types for advanced regression tasks

To handle advanced regression tasks simple linear regression is not enough, more advanced regression methods will have to be used. The models described in this section perform regression predictions in different unique ways. The models operate under the same ML basics, consisting of training and testing while measuring the error and loss of the prediction. The models perform regression prediction in different ways, but also have an optimal performance during different scenarios. This signifies the importance of selecting the correct model for the correct problem to maximize the prediction accuracy. Sekeroglu et al. (2022) compares the performance between common models with regards to regression tasks in a extensive manner and comes to the conclusion that some models perform better in terms of consistency than others. The models that are considered perform with the best and most consistent results for regression tasks are random forest, gradient boosting or extreme gradient boosting (XGBoost), support vector machine and neural networks.

Random forest

The Random forest framework operate on the basis of decision trees. The model simulates multiple decision trees that come to a prediction based on the input parameters as attributes. A common illustrative example regarding the weather could be to predict if it is going to rain or not. The input attributes are if it is a sunny sky or dark overcast. When the attribute is a dark overcast then based on that attribute the model will choose that rain is possible. This is an oversimplification but the concept is the same even with a lot of attributes (Quinlan, 1986).

Random forest is based on decision trees, while introducing an selected amount of error to the process. Breiman (2001) clarifies the inner works of the random forest model as, *"a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest"*. In more detail the process starts through bootstrapping the provided data set and creating a selected amount of decision trees based on the bootstrapped data. For the context of random forest regression bootstrapping creates new data sets for every decision tree with samples acquired from the original, resulting in a slightly different training data for every tree. Additionally during training of the trees, a random subset of features is considered at every branch split in the trees. This feature consideration is the deciding factor for if the tree should follow down one branch or another. Training the trees in this manner is a way to minimize error as the decision tree branches out. During testing every generated tree makes a prediction for x_i, where the final result is the average across all trees. Liu et al. (2012) showcases how the process for random forest prediction can be explained mathematically. The following equation describes the leaf value selection,

$$\hat{y}_t = \frac{1}{n_{leaf}} \sum_{i=1}^{n_{leaf}} y_i, \tag{2.27}$$

where the leaf is and endpoint in the decision tree. A leaf contains a subset of training data points that followed down the specific path to reach that leaf. The

value of the leaf is decided by the mean of the target values that landed in the leaf during training. This continuous process is finished once every tree have decided on a final leaf value represented as $\hat{y}_t$.

The final prediction of the random forest is calculated through taking the mean of all leaf values of all trees for a specific value of x. The final prediction in mathematical form can be described as:

$$\hat{y}_i(x_i) = \frac{1}{T}\sum_{t=1}^{T}\hat{y}_t(x_i). \tag{2.28}$$

The error measurement follows the same procedures as for linear regression with the calculation of RSS leading to a R^2 value describing the accuracy.

When it comes to practical implementation of random forest regression there are some configurations that can be made to the model. Mainly deciding on the amount of trees, max tree depth, minimum amount of samples required to create a leaf or split a branch, maximum amount of features to consider when splitting a branch and the seed for the random number generator deciding on what random features to include. Changes in these hyperparameters will affect the results of the prediction. Increasing the amount decisions the model has to make by increasing the amount of trees, branches, samples required and features to consider will require more computational power to reach a prediction. On the other hand it will provide the model with more possible leaf values, increasing the chances of a more accurate final prediction at the risk of overfitting if the data set is not large enough (Breiman, 2001). The optimal settings for the model is found through testing and reconfiguration based on the error result as the model lacks self-configuration and will have to be adjusted manually.

The random forest model is complex and the tree structure can become very large depending on setting, but the model excels when there are substantial amount of features to choose from. For smaller data sets this leads to a chance of the model not being able to interpret the relationship between features, limiting the accuracy of the final prediction (Dietterich, 1998).

XGBoost

This model is another ensemble-tree based method, meaning that the concept of multiple decision trees are used to solve a problem. What separates the random forest model and XGBoost model is the construction and utilization of the trees. On the topic of comparing random forest to XGBoost, Chen & Guestrin (2016) describes how XGBoost calculates the gradient (residual) for every tree and updates the next tree in the sequence based on the gradient from previous iterations to minimize loss. This can be seen as a sequential approach instead of the random forest parallel approach. This makes for a form of self-improvement where the results is carried over from tree to tree, continuously improving the accuracy while keeping the trees

relatively shallow. The process of sequential tree optimization is called boosting and the difference between gradient boost and XGBoost is the complexity of the algorithm and its scalability.

Nielsen (2016) explains the XGBoost algorithm as a second-order gradient boost, signifying that during the boosting part of the algorithm a second order derivative is performed to improve the loss function. This definition of the loss function is the basis for the tree construction and continuous learning leading to the end prediction made by the model. The algorithm described mathematically starts with some inputs, the data set, the number of iterations denoted as M, the the loss function L, the amount of leafs T and the learning rate, η. The learning rate controls the step size between updates of the tree.

$$\hat{f}^{(0)}(x) = \hat{f}_0 = \arg\min_{\theta} \sum_{i=1}^{n} L(y_i, \theta), \tag{2.29}$$

illustrates the initialization of the model with the constant θ serving as a starting prediction for the model.

After the first prediction has been calculated the iterative process to improve the model starts. From the provided prediction a correction for the next tree in the sequence is calculated through:

$$\hat{g}_m(x_i) = \left. \frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right|_{f(x)=\hat{f}^{(m-1)}(x_i)}, \tag{2.30}$$

and,

$$\hat{h}_m(x_i) = \left. \frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2} \right|_{f(x)=\hat{f}^{(m-1)}(x_i)}. \tag{2.31}$$

The two equations perform a first and second order derivation of the loss function with regards to the previously predicted value of the tree. the first order derivation is called the gradient and the second order the Hessian.

After the correctional gradient and the Hessian is established the tree structure can be improved. Gain, described by eq 2.32 is a measurement for how much a split in the tree improves the model. This is accomplished by adding the fraction between the sum of gradients and hessians for the left and right child nodes while subtracting the gradient and hessian of the parent nodes,

$$\text{Gain} = \frac{1}{2}\left(\frac{G_L^2}{H_L} + \frac{G_R^2}{H_R} - \frac{G_{jm}^2}{H_{jm}}\right). \tag{2.32}$$

The model learns the three structure by following the path which maximizes gain at each node.

Once the tree structure has been defined through gain, the weight of the leafs is next. $\hat{w}_{jm}$ is the weight of the j-th leaf in the m-th tree and is a result from the

gradient and hessian information for that leaf and iteration,

$$\hat{w}_{jm} = -\frac{G_{jm}}{H_{jm}}. \tag{2.33}$$

With the leaf weight defined the new tree can be fully constructed calculating a new prediction,

$$\hat{f}_m(x) = \eta \sum_{j=1}^{T} \hat{w}_{jm} \cdot \mathbb{I}(x \in R_{jm}). \tag{2.34}$$

The indicator function, $\mathbb{I}(x \in R_{jm})$ controls that x falls into the j-th leaf. The prediction is a result of the sum of leaf weights multiplied by the learning step.

After each new prediction the model must be updated,

$$\hat{f}^{(m)}(x) = \hat{f}^{(m-1)}(x) + \hat{f}_m(x). \tag{2.35}$$

Eq 2.35 adds the new tree prediction into the existing tree, resulting in a correctional update of the model for the m-th iteration.

The following equation represents the final prediction model after all iterations of boosting:

$$\hat{f}(x) = \sum_{m=0}^{M} \hat{f}_m(x). \tag{2.36}$$

The iterative boosting of the XGBoost model makes it flexible and consistent for regression tasks. Chen & Guestrin (2016) highlights the consistent positive result of XGBoost while also pointing at the risk of overfitting and underfitting when tree structures become too large or too small. It is also stated that a result due to the complex nature of XGBoost is that the model can memory-intensive when handling large data sets or large amount of boosting rounds.

Support vector regression

support vector machines (SVM) which is a model fit to handle classification tasks can be configured to instead perform regression tasks in the form of support vector regression (SVR). The objective with SVR like all regression method is to find a function that approximates the relation ship between input features and a continuous target variable. Unlike other regression models the aim of SVR is not to minimize R^2, but to instead keep it within a set error margin. The margin is denoted as ϵ_{marg} and if a prediction falls within the margin then the prediction is considered acceptable. The predictions that fall outside of the margin are the support vectors and it is the support vectors that define the regression line or curve. In other words only predictions outside of the acceptable error margin contribute to minimizing the error of the regression as they are seen as unacceptable (Smola & Schölkopf, 2004).

Two support vectors creates a tunnel for which the regression line is fitted within. The concept is illustrated in fig 2.9, showcasing how the support vectors allow the regression line to be non-linear and minimize RSS in a way that is not possible with linear regression methods.

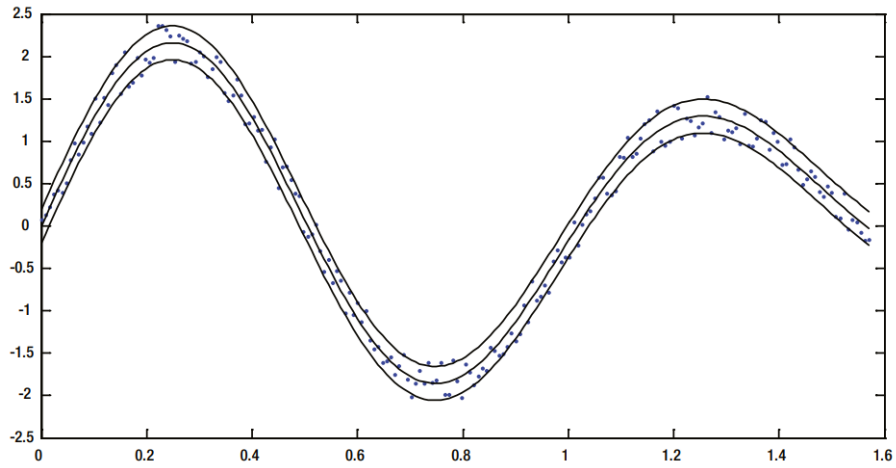


Figure 2.9: Non-linear SVR example with visualized support vectors (Awad et al., 2015)

Awad et al. (2015) presents the algorithm for SVR, starting with the linear prediction version represented by eq 2.37. The linear structure is much like the linear regression where w^T is a transposed weight vector which is the SVR method of learning through iterations.

$$f(x) = w^T x + b \tag{2.37}$$

Going from linear to non-linear means that there must be more controls for the support vectors above and below the error to guide the regression line in an accurate manner. The following equation,

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x) + b, \tag{2.38}$$

showcases that the non linearity can be predicted by utilizing a non-linear kernel function in place of x. Lagrange multipliers α_i and α_i^* are introduced in place of w^T to penalize prediction values outside of the error margin.

The concept of introducing a kernel is commonly called the "kernel trick", and it allows the SVR to be configured for different types of data. There are many available kernels ranging from linear, polynomial to radial basis function kernels. The kernel trick, visualized in fig 2.10 is the process of transforming the original input data into higher dimensional space. This space is referred to as the kernel space and this transformation provides SVR with a lot of flexibility to handle problems for multiple

types of data. To transform the input space into the kernel space a mapping function is needed, often denoted as $\varphi(x)$. In the new space the previously complicated and inseparable data can be separated by a linear hyperplane, which creates an environment where linear optimization is possible.

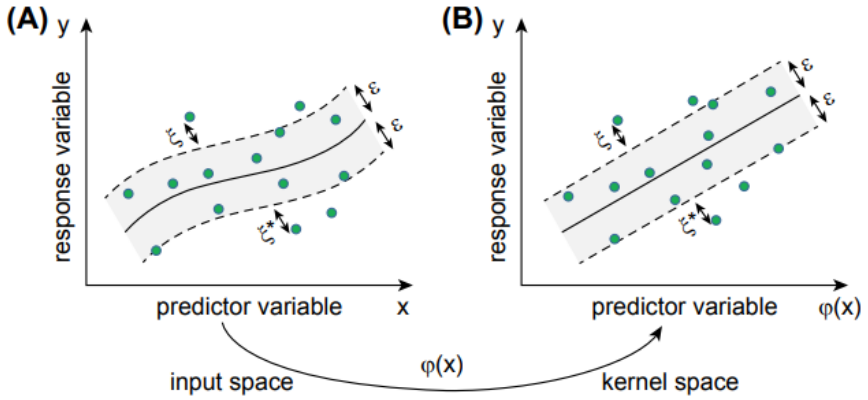


Figure 2.10: Graphic illustration of transforming data from an input space into a kernel space (Zhang & O'Donnell, 2020)

Zhang & O'Donnell (2020) delves deeper into the optimization of $f(x)$ through the following equation:

$$\min_{w,\xi_i,\xi_i^*} \left(\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \right),$$

$$\begin{aligned} &\text{subject to,} \\ &y_i - f(x_i) \leq \varepsilon + \xi_i, \\ &f(x_i) - y_i \leq \varepsilon + \xi_i^*, \\ &\xi_i, \xi_i^* \geq 0. \end{aligned} \tag{2.39}$$

It is stated that in a non linear setting this optimization problem is to solve the flatness in the kernel space, not in the input space.

The slack variables ξ and ξ^* can be spotted in fig 2.10, and determine how many data points outside the margin that will be tolerated. Determination of trade-off between flatness of the function f and the prediction errors is done through the regularization parameter C. A large C provides more weight to minimize prediction errors, and a small C more weight to minimizing the function flatness (Cortes & Vapnik, 1995).

By representing eq 2.39 in terms of support vectors, the problem is transformed into a dual optimization problem. The equation and simplification:

$$\max_{\alpha,\alpha^*} \quad -\frac{1}{2}\sum_{i\in SV}\sum_{j\in SV}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)K(x_i,x_j)-\varepsilon\sum_{i\in SV}(\alpha_i+\alpha_i^*)+\sum_{i\in SV}y_i(\alpha_i-\alpha_i^*),$$

where $K(x_i, x_j) = \varphi(x_i)\varphi(x_j)$, is subject to,

$$\sum_{i\in SV}(\alpha_i-\alpha_i^*)=0, \alpha_i, \alpha_i^* \in [0, C], \tag{2.40}$$

clarifies the dual optimization problem and the path towards support vector definition through a sum of the two Lagrange multipliers.

This makes it possible to represent w as a form of summation between Lagrange multipliers and the mapping function,

$$w = \sum_{i\in SV}(\alpha_i-\alpha_i^*)\varphi(x_i). \tag{2.41}$$

With new definitions for weight and x, eq 2.37 can be adjusted into a non linear form eq 2.38 (Zhang & O'Donnell, 2020).

SVR which is built upon the principles of SVM is a mathematically complicated model with great regression problem solving capabilities. The model excels compared to other models in predicting continuous outcomes due to the implementation of kernels. The main area of possible configuration lies in kernel selection, choosing a kernel which corresponds to the structure of the input data will yield better results. Other possible configurations are error margins and the regularization parameter to adjust penalties (López et al., 2022).

Neural networks

The inspiration for the complex structure of neural networks (NN) is the biological structure of the human brain. It possesses an in many cases unmatched information-processing power, being able to handle multiple types of classifications and regression with incredible speed and accuracy. López et al. (2022) provides an example for the brains processing power regarding the human vision. The vision organs gather visual information regarding surroundings at high speeds and when combined with the brain information is given as to how one can interact in different scenarios and situation, such as identifying danger or recognizing people. Perceptual recognition is the human brains strongest point, combining the human sense makes for a system that is capable of accurate predictions in approximately 200 ms. For computational methods even achieving lesser tasks takes much longer, signifying the power of the human brain.

The closest computational model compared to the brain that is available in current day is NN or artificial neural networks (ANN). A NN utilizes a structure of neurons

to make predictions, hence the comparison with the human brain. According to Zou et al. (2009) each individual neuron in a network operates on the basis of an activation function where the input x comes from previous neurons or the input features and the weights from the connections between neurons. When the weighted sum of inputs passes a certain threshold then the activation function activates and sends the signal to neighboring neurons or to the output.

The neurons are connected together in layers starting with one input layer, a configurable amount of hidden layers and one output layer. Fig 2.11 illustrates how a typical NN is structured with visualized connections between neurons. This form of structure is highly customizable where the amount of hidden layers as well as the amount of neurons per layer can be configured. An increase in layers and neurons requires more processing power but will be able to grasps relationships between features more successfully, and wise versa a smaller network makes predictions faster and might therefore be better suited for smaller data sets (Rafiq et al., 2001).

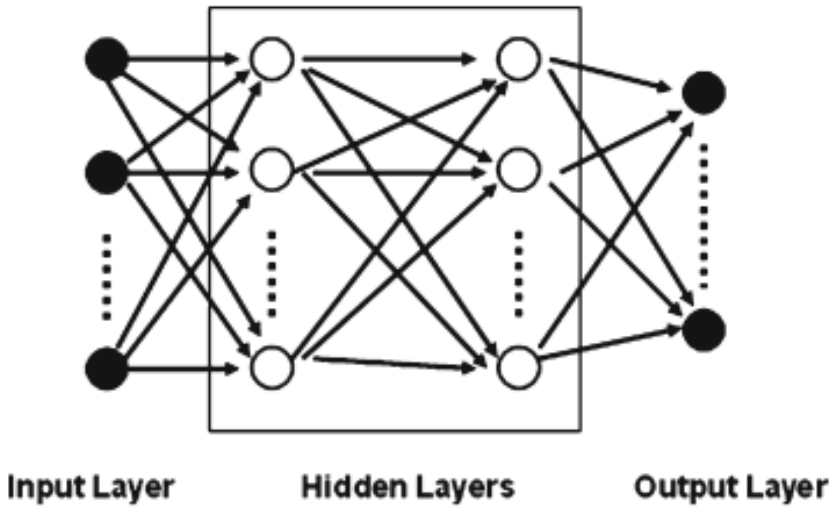


Figure 2.11: Neural network layer structure (Zou et al., 2009)

With the topology of the network defined the process of generating a prediction can begin. The input layer provides a starting point of neurons that send signals forward to the next layer in a continuous process until the output layer is reached. Weights and biases are added for every neuron to adjust the values as they get send onward. This process is called forward propagation and is the prediction phase of the NN model. After predictions the NN model works backwards comparing the prediction to the actual value. This allows the model to adjust weights between neurons to achieve a better prediction during the next iteration, this process is called back-propagation and is the learning phase for a NN (Dreiseitl & Ohno-Machado, 2002).

To deeply explain the workings of a NN the firs building component is the neuron itself. Ketkar et al. (2021) in the context of creating a NN in Python refers to:

$$z_{ik}^{(h)} = \sum_{p=1}^{P} x_{ip} w_{kp} + b_j^{(h)}, \tag{2.42}$$

as the mathematical function of a neuron in the hidden layer. The function takes the sum of all input values (x_{ip}) and weights (w_{kp}) from the previous layer of input neurons (P). The scalar $b_j^{(h)}$ is an additional trainable parameter which is specific for the single neuron and provides the neuron with adaptability to better fit the function. This scalar can be represented as shown in eq 2.42 or can be circumvented for easier mathematical representation with an extra input neuron with a set value of 1. P represents the total amount of neurons in the input layer/ previous layer connecting to the k-th neuron in the hidden layer.

Assuming the circumvention of the bias López et al. (2022) unravels the equations for forward- and back propagation from the sum of inputs into the hidden layer ($z_{ik}^{(h)}$) it is possible to establish the output of the k neuron through adding a activation function, denoted as g^h and shown as,

$$V_{ik}^{(h)} = g^h(z_{ik}^{(h)}). \tag{2.43}$$

$V_{ik}^{(h)}$ is the output from a neuron k within the hidden layer during epoch i.

To continue and calculate the values generated by neurons in the output layer a similar structure as eq 2.42 is used with the input values being the calculated values established from eq 2.43. The result is the sum of values being transferred to the output layer as defined in the following equation:

$$z_{ij}^{(l)} = \sum_{k=1}^{M} w_{jk}^{(l)} V_{ik}^{(h)}. \tag{2.44}$$

The very final step to get the prediction values from the model is to apply the output layer activation function,

$$\hat{y}_{ij} = g^{(l)}(z_{ij}^{(l)}). \tag{2.45}$$

Rafiq et al. (2001) states how there are different versions of NN which have different versions of neuron and activation function definitions. The different versions excel at different things but the most common and recommended version of NN is multi-layer perceptron (MLP). MLP operates on the principles described above, meaning the incorporation of forward and backward propagation.

With the forward propagation step defined the backpropagation is the second part in a NN operating on MLP principles. Chauvin & Rumelhart (2013) describes the process in multiple mathematical steps starting with the definition of the gradient

of the loss function. The functioned used is a sum of squared error function (SSE), which shares many similarities with a RSS function and is represented as:

$$E = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{L}(\hat{y}_{ij} - y_{ij})^2. \tag{2.46}$$

The loss function is iterated over the training samples (n) and the number of output neurons (L) to get the optimal weight values.

Utilizing the negative gradient of the SSE function while multiplying in the learning rate (η) the change in weights for the hidden layer can be defined as,

$$\Delta w_{jk}^{(l)} = \eta \frac{\partial E}{\partial w_{jk}^{(l)}}. \tag{2.47}$$

Expanding eq 2.47 with the chain rule and solving each partial derivative,

$$\Delta w_{jk}^{(l)} = \eta \frac{\partial E}{\partial \hat{y}_{ij}} \frac{\partial \hat{y}_{ij}}{\partial z_{ij}^{(l)}} \frac{\partial z_{ij}^{(l)}}{\partial w_{jk}^{(l)}}, \tag{2.48}$$

results in a detailed equation for the change in weights from the hidden layer to the output layer:

$$\Delta w_{jk}^{(l)} = \eta \left(y_{ij} - \hat{y}_{ij}\right) g^{(l)'}\left(z_{ij}^{(l)}\right) V_{ik}^{(h)} = \eta \delta_{ij} V_{ik}^{(h)}. \tag{2.49}$$

Where $\delta_{ij} = \left(y_{ij} - \hat{y}_{ij}\right) g^{(l)'}\left(z_{ij}^{(l)}\right)$. With all the components calculated an equation representing the formula used to update weights from the hidden layer to the output layer is defined,

$$w_{jk}^{(l)(t+1)} = w_{jk}^{(l)(t)} + \eta \delta_{ij} V_{ik}^{(h)}. \tag{2.50}$$

With eq 2.50 finalized the weights in between output layer and hidden layer have been updated to better match the provided data as well as to make better predictions. López et al. (2022) states how the weights between hidden layer and input layer also must be updated and is done in a similar manner with some changes represented by the calculations below.

Firstly the change in weights between input and hidden layer is defined,

$$\Delta w_{kp}^{(h)} = \eta \frac{\partial E}{\partial w_{kp}^{(h)}} \tag{2.51}$$

The change in weights are then expanded and partial derivatives calculated. The expansion represented by:

$$\Delta w_{kp}^{(h)} = \eta \sum_{j=1}^{L} \delta_{ij} w_{jk}^{(l)} g^{(h)'} \left(z_{ik}^{(h)} \right) x_{ip} = \eta \psi_{ik} x_{ip}, \tag{2.52}$$

where $\psi_{ik} = \sum_{j=1}^{L} \delta_{ij} w_{jk}^{(l)} g^{(h)'} \left(z_{ik}^{(h)} \right)$,

is the partial derivations from eq 2.48 with the extra steps to reach the weights between the input and hidden layer. The addition of the previous partial derivatives is why the process is called backpropagation, due to the weights being adjusted in the opposite direction compared to the predictions.

Just like the previous update (eq 2.50) the update formula is applied in the same manner resulting in:

$$w_{kp}^{(h)(t+1)} = w_{kp}^{(h)(t)} + \eta \psi_{ik} x_{ip}, \tag{2.53}$$

updating the weights between input and hidden layer. The whole process of back-propagation and changing values of weights continues until the end of all epochs, continuously improving the model for every epoch. Hecht-Nielsen (1992) provided early documentation of the process of backpropagation and since the process has been improved in many different ways. The concept is still the same and it is stated that the process provides immense learning capabilities with much flexibility for understanding the input features.

Dreiseitl & Ohno-Machado (2002) provides further insight into the configuration of NN. It is stated that a NN runs significant chances of overfitting if the topology of the network, meaning hidden layers and neurons, is not matched with the amount of input features and data set. For smaller data sets a complex topology will result in an increase in required processing power with a large chance of overfitting. For a large data set with many input features a small network will result in the NN not being able to grasp relationships resulting in an unsatisfactory prediction.

2.2.3 Feature engineering in propeller design

In previous statements it has been assumed that the data set for the task is good, in many cases however the data set must be modified to achieve the best possible results. Modification of data to better fit the task such as dividing functions into their core components instead of the final function value is called feature engineering. Dong & Liu (2018) highlights the importance of feature engineering and how good configuration of the data set is much more important than the correct settings for a ML model.

For creating a data set with regards to propeller design and ML many methods are available depending on the end goal of the prediction. Doijode et al. (2022) defines

the geometric profile of a propeller blade as b-spline curve with the values of the curve represented in a matrix with the goal of optimizing propeller performance. Tadros et al. (2024a) provides an additional example of feature engineering where ship characteristics and general propeller parameters such as EAR and advance coefficient are used to generate a propeller geometry from a surrogate model. These example indicate how propeller geometry can be represented in multiple ways as input features, but it is stated that organizing the data set to ease the learning process, e.g, utilizing numerical data with a direct connection to the output, is recommended.

2.3 Previous work

Many studies have historically been conducted on the subject of both ML and propeller design. In this section a selection of previous works will be showcased where the main focusing utilizing ML and propeller design in combination.

Doijode et al. (2022) describes the utilization of unsupervised classification methods to optimize propeller design with regards to efficiency and cavitation. A large portion of their methodology aims to improve the computational cost that is required for the ML models to generate good results. The decision to use multiple clusters with different goals to get an final design cluster with high performance provides insight for how ML can be divided into sub-tasks to fulfill the final goal.

Gypa et al. (2024) in a study regarding the utilization of different optimization algorithms highlights the usage of MLP models to improve propeller geometry to increase performance. The methodology involves cross validation and grid search techniques to find optimal hyperparameters for the model. The study involves propeller designers into the methodology as a form of interactive optimization. Additionally the paper offers valuable input as for how a ML pipeline can be designed for the task of improving propeller design in an optimized way.

The choice of input parameters for a ML model is a critical step for the models performance. Pan et al. (2023) delves deep into the effect of common propeller design parameters and their relationship as inputs and outputs. The study indicates the importance of D, EAR, K_T, K_Q, the number of blades and the RPM as input parameters for ML models working on propeller design tasks. The different variables are believed to hold the largest significance for explaining variance between the variables and the geometrical output.

3

Methodology

In this chapter the methods used during different steps of the thesis project will be presented. The methods will be presented in a chronological order to ease understanding of the process. More details regarding full code scripts and related information that is not presented in this chapter can be found in the appendix.

3.1 Propeller dataset collection

The collection of data was done in collaboration with *Berg Propulsion* which provided access to a library of in-house propeller designs. The designs consisted solely of conventional shaftline propellers, serving as the main propulsion system for the vessel. The data for the propeller designs had been calculated through an in-house calculator which calculates blade geometry and other parameters based on a set of inputs.

The design files where all converted into .csv files for processing purposes and sorted based on the data present in each file. The current version of the calculator was also provided with the purpose of data creation and experimentation.

The collected data files contained the full propeller geometry, data used as input into the original calculation and performance curves for each propeller. The data in each file was studied and checked to be in a similar enough format to be used for the task. In the case of necessary parameters being missing from a design, the current version of the internal calculator was used to roughly correct the data. 58 propellers was deemed as usable for the study and was further refined.

3.2 Feature representation & pre-processing

The full geometric profile of a singular propeller blade consist of 529-591 data points depending on size, describing the shape of the blade across radial sections. Simplifications of the geometrical data was done to avoid using the total amount of data points. The data was simplified to consist of the propeller blades P/D values, Length from center of the chord line to the leading edge (LLE), length to the trailing edge (LTE) and camber. The camber in this instance is mapped as a ratio along the

total length of the chord (L_{tot} and the distance to the camber line defined by center between the suction- and pressure side of the blade. Camber ratio is measured at the center of the chord length for all radial sections for simplification.

The parameters were mapped across 11 radial sections meaning that the full set of outputs amount to 44 data points for each individual propeller. The 44 data points were used as a base dataset containing no modifications and arranged in a flattened array for accessibility. In the case the case of gaps in data such as missing values, then the provided calculator was firstly used to get a reference value and then an estimated value was used based on design expert input and knowledge. Radial section values were equal for all propeller blades, meaning that every data point for each propeller was plotted against the same sections and no variation was present in terms of sections.

The input data was selected based on the literature study and professional input. Table 3.1 is the input parameters chosen for each propeller design. The input data was also structured as a simple array for simplicity and with the purpose of potential reconfiguration.

Table 3.1: Table of input parameters used for the ML models.

Input Parameters	Unit
Advance coefficient	-
Thrust	N
Torque	Nm
Rotations per minute	RPM
Diameter	mm
expanded area ratio	%
Skew	∘
Cavitation margin	%
Rake	∘
Ship speed	kt
Delivered power	kW

3.3 ML model design

With the input and output data defined some initial parameters was chosen. Firstly by deciding the test and training split as 80/20, meaning that training was done on 80% of the data set and testing on the remaining 20%. The inputs was standardized before being used by the models by utilizing `StandardScaler` from the package `sklearn.preprocessing`. The data was then fully ready to be used by the models.

The machine learning algorithms chosen for the task were, Random Forest, XGBoost, NN and SVR. After selection of models the design of models was done through the help of the libraries `sklearn` library for Random forest, NN and SVR while `xgboost` is its own separate library specifically for XGBoost. Every library

and model had its own set of configurable parameters, which regulated model specific performance and settings. To not iterate through multiple settings to find the best configuration the function `GridSearchCV` from the library `sklearn` was used. The parameters for every model along with the variation in parameter values is represented by the following tables,

Table 3.2: Grid search parameters used for Random Forest

Random Forest	
n_estimators	100, 300, 500
max_depth	10, 20, 30, None
min_samples_split	2, 5, 10
min_samples_leaf	1, 2, 4
max_features	'sqrt', 'log2'

Table 3.3: Grid search parameters used for XGBoost

XGBoost	
n_estimators	500, 800, 1000
max_depth	5, 10, 15
learning_rate	0.01, 0.05, 0.1
subsample	0.6, 0.8, 1.0
colsample_bytree	0.6, 0.8, 1.0

Table 3.4: Grid search parameters used for MLP Regressor

MLP (Neural Network)	
Maximum iterations	1000, 2000, 3000
hidden_layer_sizes	(64, 64, 32), (128, 64), (100,)
activation	'relu', 'tanh'
alpha	1e-4, 1e-3, 1e-2
learning_rate_init	0.001, 0.005, 0.01

Table 3.5: Grid search parameters used for Support Vector Regression

SVR	
kernel	'linear', 'rbf', 'poly'
degree	2, 3, 4 (if 'poly')
C	0.1, 1.0, 10.0
epsilon	0.01, 0.1, 0.2

When settings involved a random state, then the default value 42 was chosen for the sake of consistency. With the grid search in place, all models were fully configured and tested on the inputs.

3.4 Evaluation metrics

The `GridSearchCV` function included a cross-validation algorithm which was set to perform 10 folds for every model. The cross-validation and optimization of model parameters results was based on the highest R^2 score.

With the final predictions done MSE, MAE, EVS and R^2 was calculated for the model as a whole and also for each individual output. The evaluation metrics was plotted in the form of a bar chart displaying R^2 for each individual output, a flattened scatter plot between predicted and actual values and finally a scatter plot between residuals and predicted values.

Based on evaluation metric results throughout iterations modifications to the data set or models was done to increase the accuracy. This consisted of tuning the grid search parameters to include more variations for comparison or removing parameters that were not useful to make predictions faster.

4

Prediction Results

This chapter will display all results in the form of prediction scores, plots, and optimal configurations for models. First an overlook of mean results between models are presented, which is followed by detailed performance for each individual model and parameter. More detailed results for individual parameter performance are found in Appendix A.

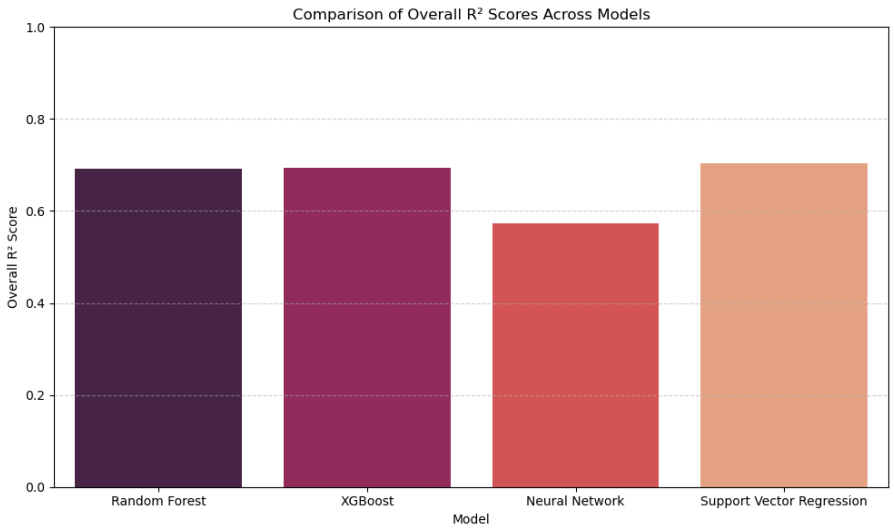


Figure 4.1: Overall R^2 score between models

In the general scoring all models except for neural networks perform with the same accuracy at aproximately $R^2 = 0.7$.

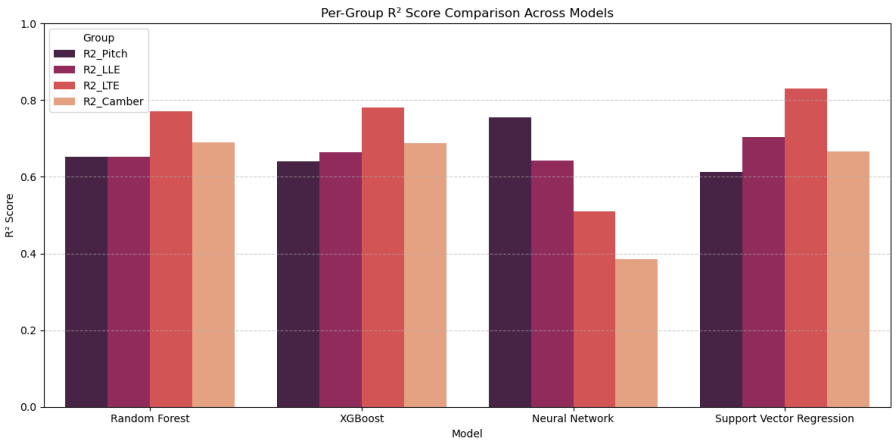


Figure 4.2: Overall parameter R^2 score between models

In a more detailed setting, random forest XGBoost perform in a almost identical manner with only a slight difference in general pitch accuracy. The NN model has the highest performance in terms of general pitch R^2 score, reaching a $R^2 = 0.77$. The LLE parameter is on the same level as the rest with LTE and camber on the lower end reaching R^2 score of 0.5 and 0.38 respectively. SVR is on par with XGBoost and random forest while achieving the highest R^2 score for LTE at 0.82.

4.1 Random forest predictions

Table 4.1: Optimal random forest parameters

n_estimators	500
max_depth	20
min_samples_split	2
min_samples_leaf	1
max_features	'sqrt'

Table 4.2: Overall performance metrics for random forest model

Measurement	Value
MAE	161.4846
EVS	0.7152
MSE	110278
R^2	0.6915

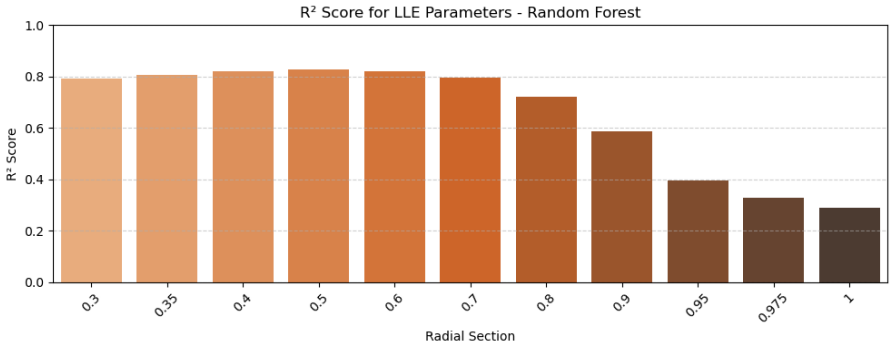


Figure 4.3: R^2 scores across radial sections for length to leading edge with random forest.

For LLE across the radial sections a pattern is found where the model is performing considerably higher in the earlier radial sections at an R^2 score of above 0.8, but at later sections the score drops down to around 0.3.

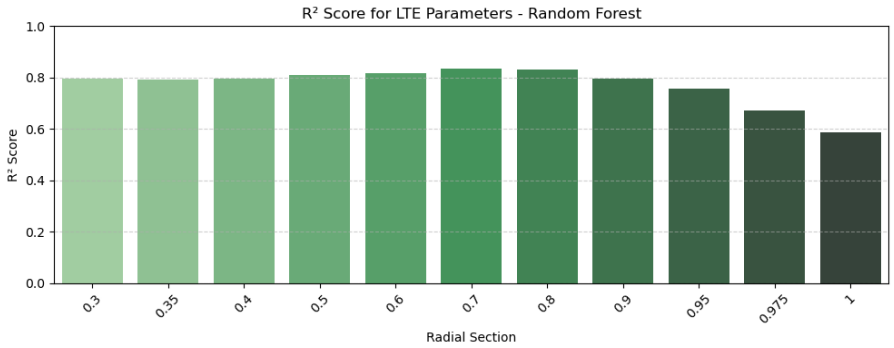


Figure 4.4: R^2 scores across radial sections for length to trailing edge with random forest.

For LTE the score stays consistently over 0.8 until radial section 0.95 where the accuracy starts to drop reaching just below 0.6 at the tip of the blade.

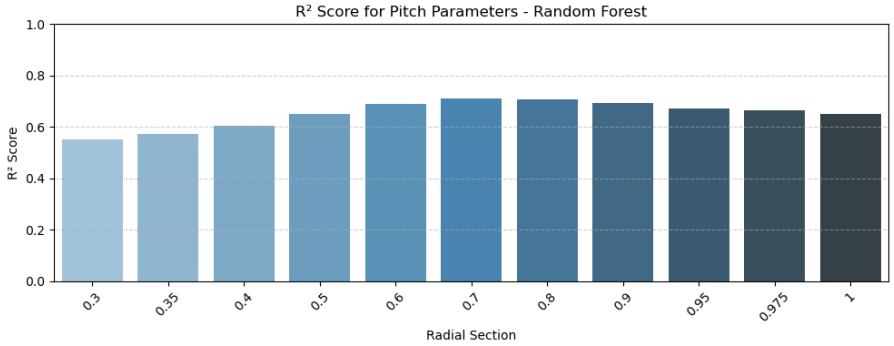


Figure 4.5: R^2 scores across radial sections for pitch with random forest.

The R^2 score for pitch lies just below 0.6 in earlier radial sections peaks at 0.7 in the middle sections and eventually evens out at just above 0.6 at the tip of the blade.

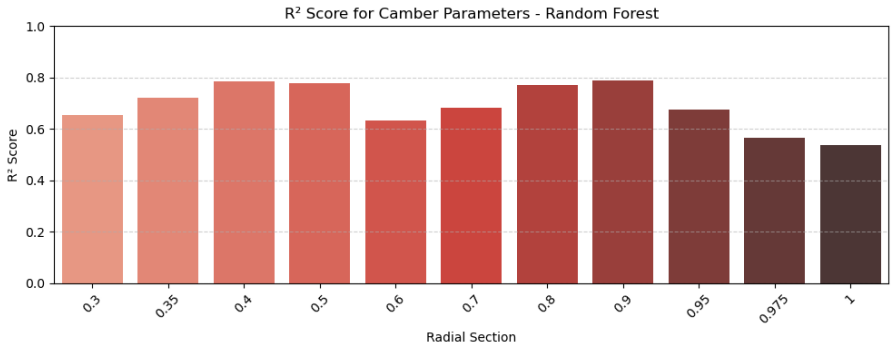
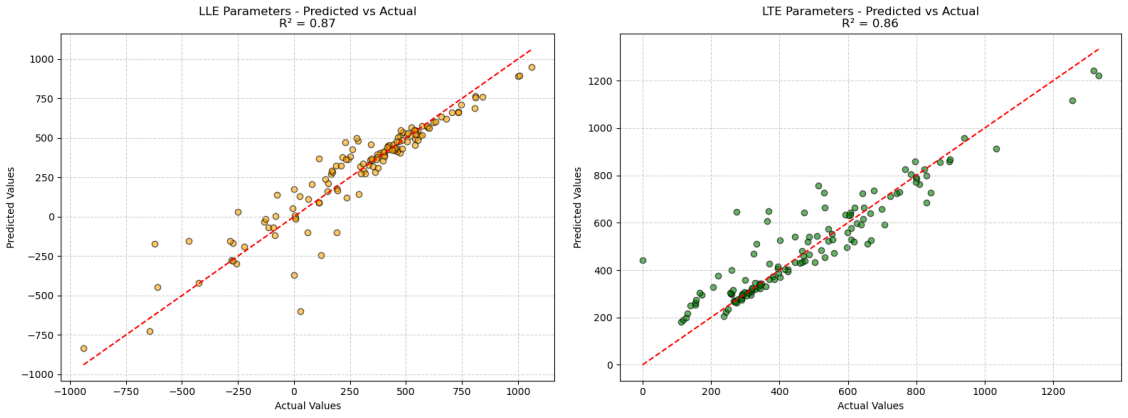
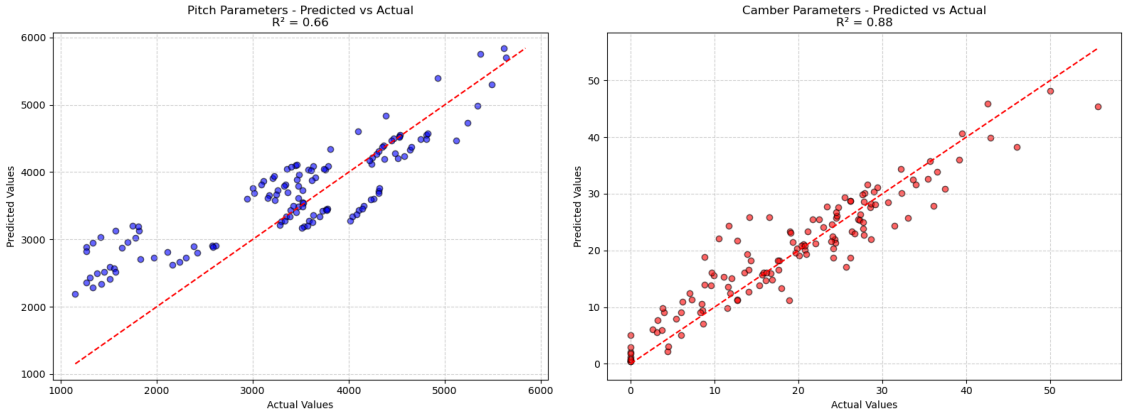


Figure 4.6: R^2 scores across radial sections for camber with random forest.

For camber the R^2 score has two observable local maxima points with a dip in R^2 score in the middle sections. The score ranges between approximately 0.8 to 0.68.



(a) LLE **(b)** LTE



(c) Pitch **(d)** Camber

Figure 4.7: Random Forest plots between predicted and actual values for each parameter

The plots between predicted and actual values showcases that for LLE and LTE that the predicted values tend to stay relatively accurate with a few outliers for when

the actual value is reaching towards 0 or into negative values. Pitch and Camber are more spread out with a consistent diversion from the actual values

4.2 XGBoost predictions

Table 4.3: Optimal XGBoost parameters

XGBoost	
n_estimators	500
max_depth	5
learning_rate	0.05
subsample	0.6
colsample_bytree	0.6

Table 4.4: Overall performance metrics for XGBoost model

Measurement	Value
MAE	160.169
EVS	0.7195
MSE	113606.1632
R^2	0.6940

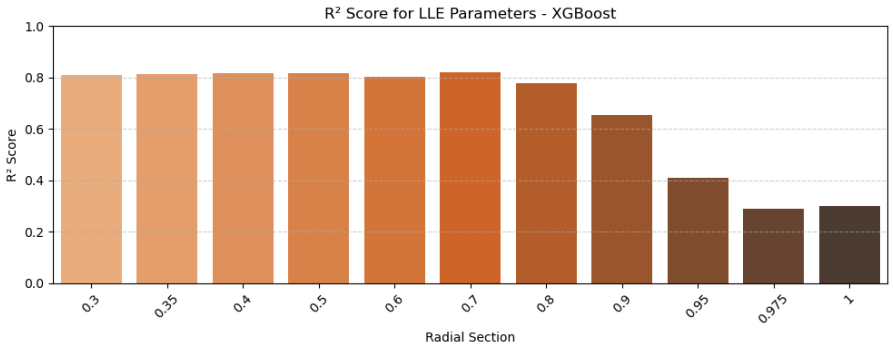


Figure 4.8: R^2 scores across radial sections for length to leading edge with XGBoost.

Score follows the same pattern as random forest with a stable R^2 score for earlier radial sections and a dip towards the tip going from a R^2 score of 0.8 to 0.3.

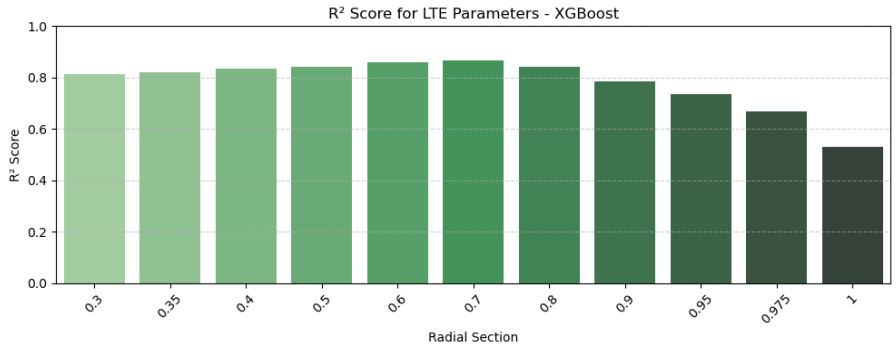


Figure 4.9: R^2 scores across radial sections for length to trailing edge with XGBoost.

Follows an almost identical pattern as fig 4.4 in terms of score values and pattern

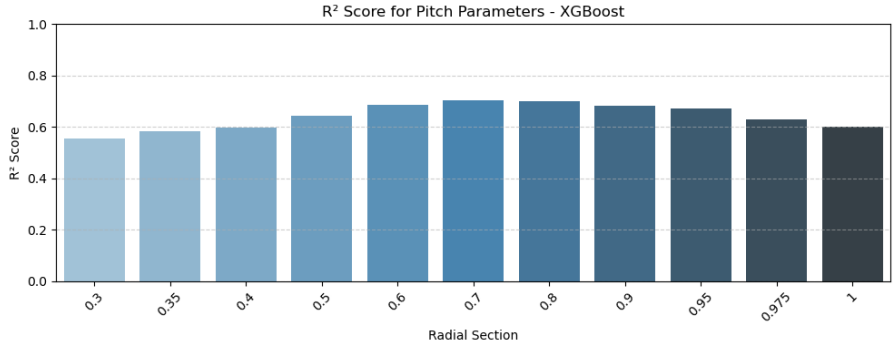


Figure 4.10: R^2 scores across radial sections for pitch with XGBoost.

Same situation for Pitch, with the R^2 score starting at 0.58 in the first radial section and climbs up to 0.7 before going back dow to 0.6.

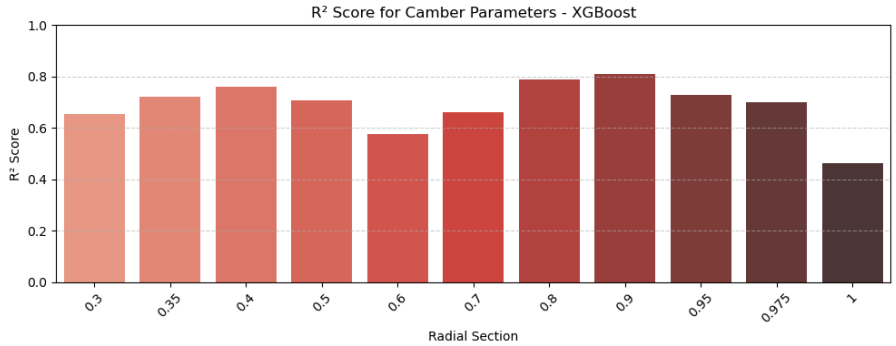
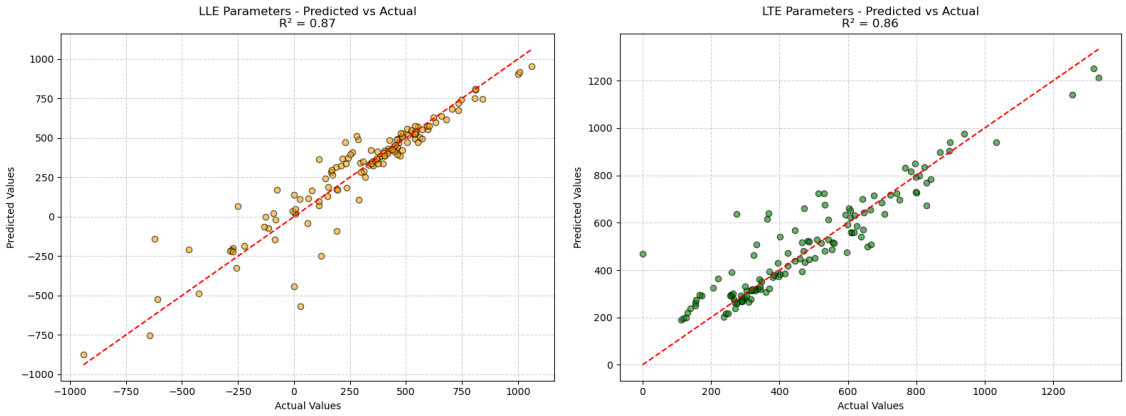
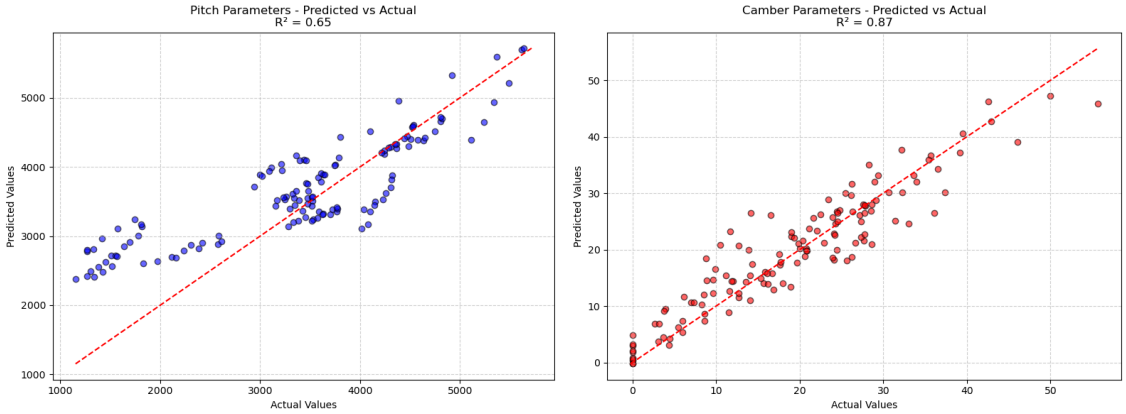


Figure 4.11: R^2 scores across radial sections for camber with XGBoost.

Camber also follows the same pattern as random forest ranging from R^2 score 0.45 to 0.8 with a local minima at radial section 0.6.



(a) LLE **(b)** LTE



(c) Pitch **(d)** Camber

Figure 4.12: XGBoost plots between predicted and actual values for each parameter.

The actual values compared to the predicted values follows the same structure as the random forest model with increased uncertainty in results for LLE and LTE when values reaches towards or go below 0. Pitch and camber are spread out across all compared values with pitch especially forming small clusters while camber is the most sporadic parameter.

4.3 Neural Network predictions

Table 4.5: Optimal NN MLP parameters

MLP (Neural Network)	
Max iterations	1000
hidden_layer_sizes	(128, 64, 32)
activation	'relu'
alpha	0.01
learning_rate_init	0.005

Table 4.6: Overall performance metrics for NN MLP regressor model

Measurement	Value
MAE	145.0363
EVS	0.6179
MSE	83107.8334
R^2	0.5734

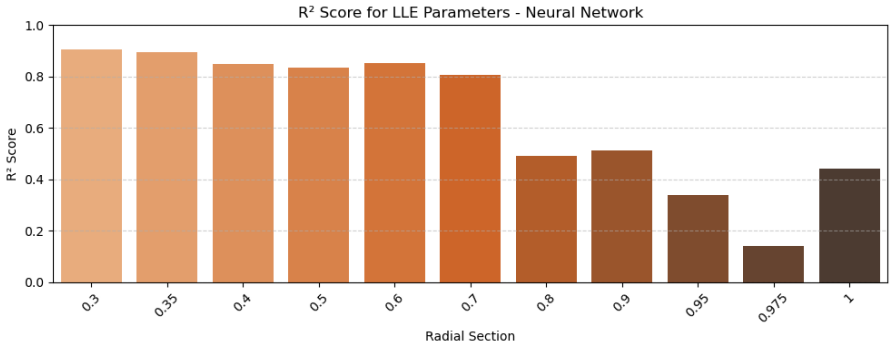


Figure 4.13: R^2 scores across radial sections for length to leading edge with NN.

The LLE predictions for neural network ranges, in R^2 score, from 0.9 down to 0.14 with the higher range at earlier radial section and a big drop in score towards the tip.

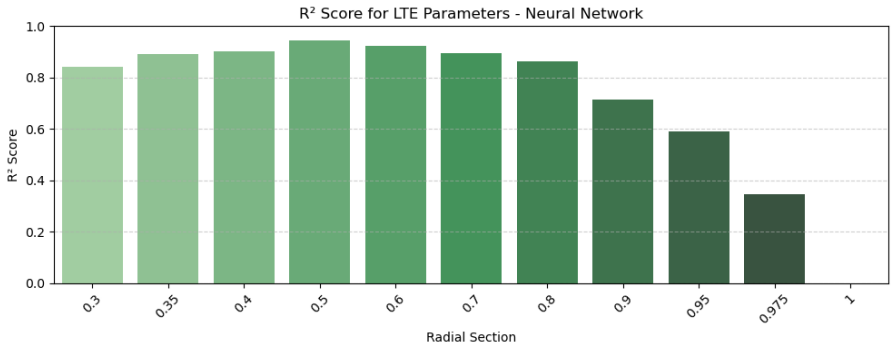


Figure 4.14: R^2 scores across radial sections for length to trailing edge with NN.

The R^2 score for LTE peaks at radial section 0.5 with a score of 0.92. The score then steadily dips downwards until radial section 1 where the value is negative and therefore performs lower than the mean value of actual values.

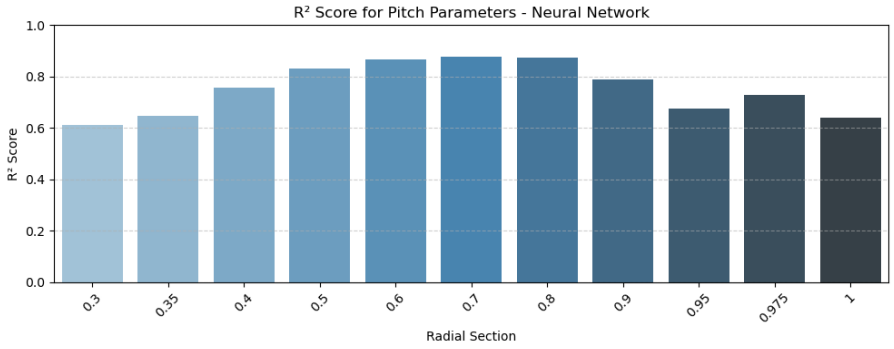


Figure 4.15: R^2 scores across radial sections for pitch with NN.

Pitch scores ranges from 0.6 to 0.84 at the local maxima at radial section 0.7 towards the tip scores are still above 0.6 but varies a lot despite the small distance from radial section 0.95 to 1.

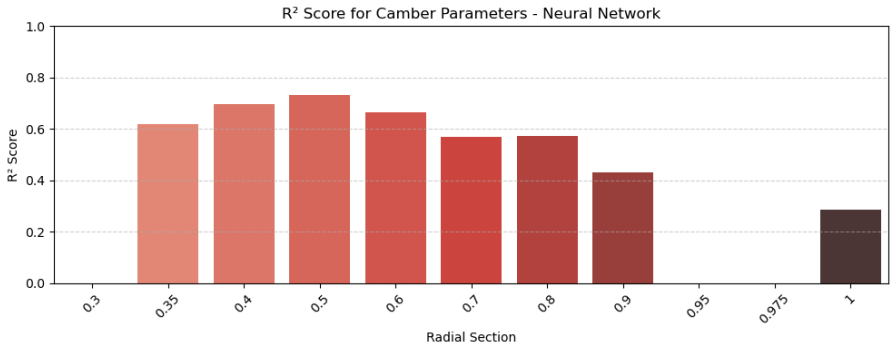
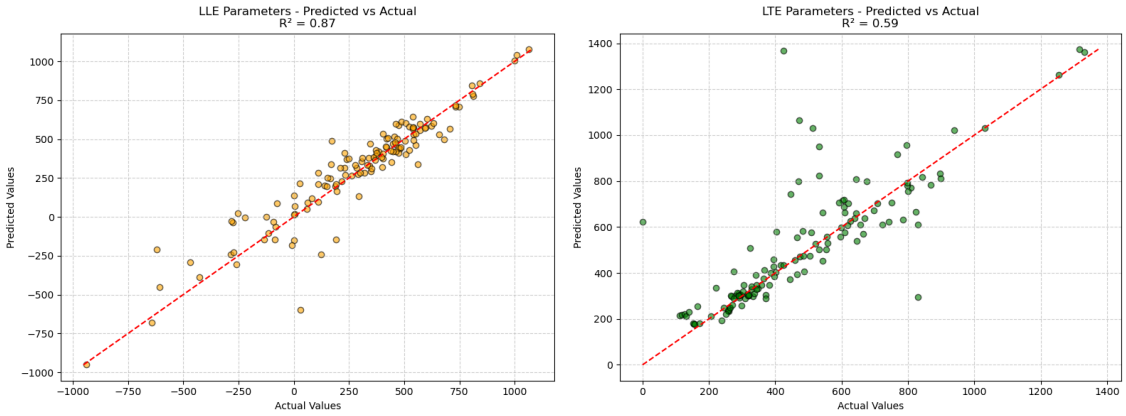


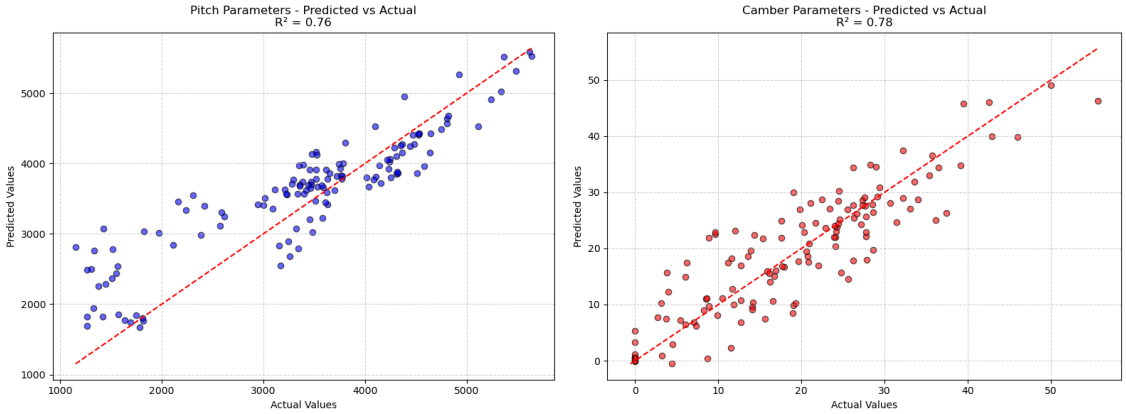
Figure 4.16: R^2 scores across radial sections for camber with NN.

Scores for Camber are very sporadic reaching into negative for radial sections 0.3,0.95

and 0.975. Other radial sections achieves the R^2 score range between 0.7 at the highest down to 0.3 at the lowest.



(a) LLE **(b)** LTE



(c) Pitch **(d)** Camber

Figure 4.17: Neural Network plots between predicted and actual values for each parameter

Patterns for actual vs predicted values follows approximately the same characteristics as the other models but noticeable for LTE is that a few predicted values divert far from the actual values this can also be seen in the camber plot where the diversion is larger constantly for all values compared to other models.

4.4 Support vector regression predictions

Table 4.7: Optimal SVR parameters

SVR	
kernel	linear
degree	2
C	10.0
epsilon	0.2

Table 4.8: Overall performance metrics for SVR model

Measurement	Value
MAE	166.5979
EVS	0.7186
MSE	121498.0156
R^2	0.7031

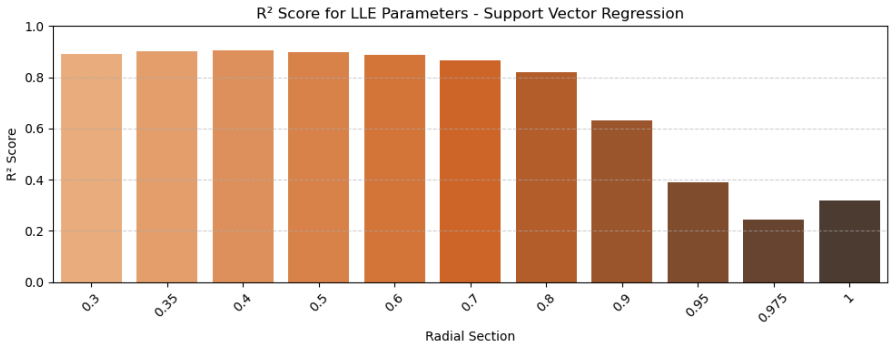


Figure 4.18: R^2 scores across radial sections for length to leading edge with SVR.

For the SVR model scores are in general close in resemblance to the random forest and XGBoost models in terms of pattern but the ranges are slightly different. For LLE the higher scores in earlier section lie at approximately 0.9 until section 0.7 where the scores dip steadily downwards, reaching 0.3 at the tip of the blade.

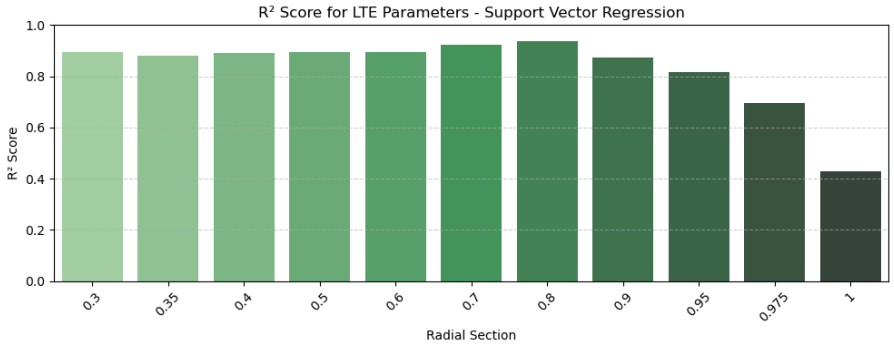


Figure 4.19: R^2 scores across radial sections for length to trailing edge with SVR.

LTE follow the same score pattern but with the highest achieved score at radial section 0.8, being 0.92 before dipping down to a score of 0.41.

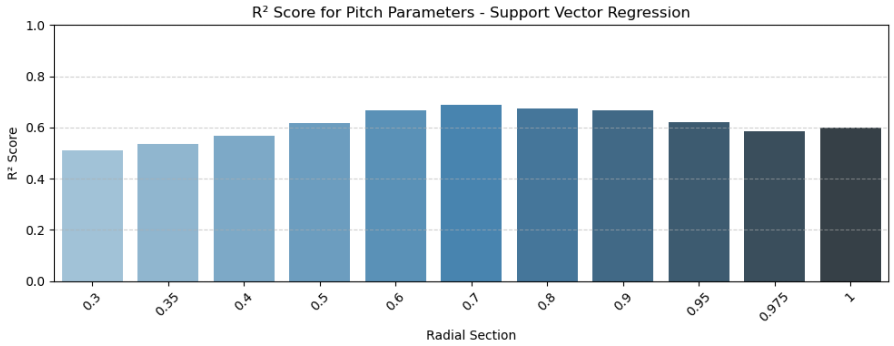


Figure 4.20: R^2 scores across radial sections for pitch with SVR.

The score for pitch is relatively even ranging from 0.52 at its lowest to 0.68 at its peak which is found at radial section 0.7.

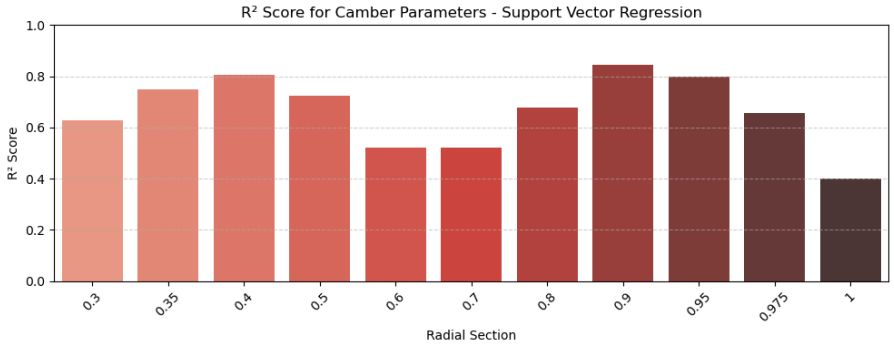
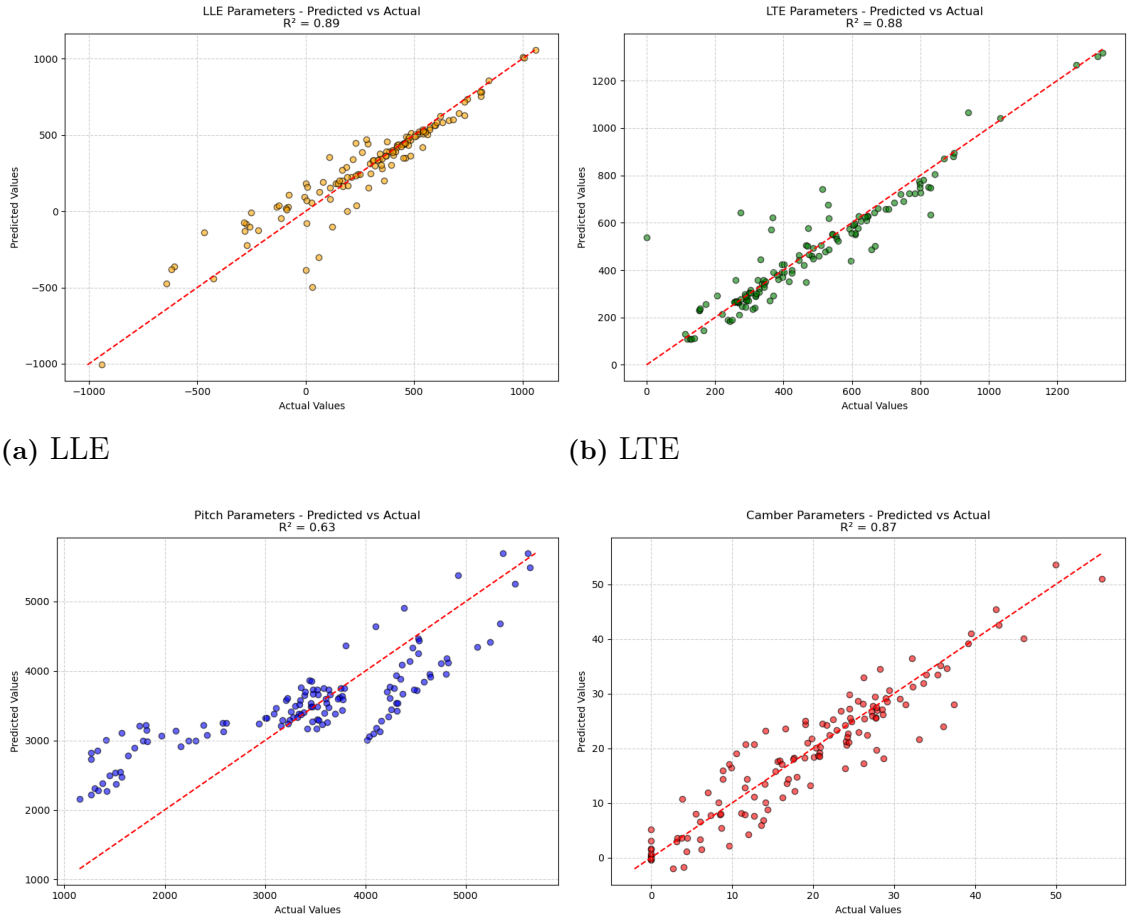


Figure 4.21: R^2 scores across radial sections for camber with SVR.

Camber follows again the same pattern as other models with two local maximum points and a local minimum point at radial section 0.7. R^2 score in general ranges from 0.52 to 0.82.



(a) LLE **(b)** LTE

(c) Pitch **(d)** Camber

Figure 4.22: Support vector regression plots between predicted and actual values for each parameter

Comparing the actual values to the predicted value for LLE and LTE it can be observed that the SVR model has relatively little diversion when compared to other models as most of the values match up. For camber and pitch, they follow the same pattern as other models being less accurate in their predictions with pitch forming clusters and the camber values also forming clusters at certain values. These values are noticeably at 0 and 28 where the model makes the best predictions.

5

Discussion

Here the results and project as a whole will be discussed from multiple standpoints with the goal of providing a sufficient explanation of how the results came to be, why the results hold significance and potential improvements for the future.

5.1 Interpretation of results

When comparing the models overall performance metrics it can be observed that the SVR model has a higher performance in terms of R^2- and EVS score. The model is scoring particularly high on LLE and LTE parameters, but has lower results in pitch and camber predictions when compared to other models. This variation in prediction performance explains the higher values in terms of MAE and MSE, indicating that in comparison the model explains variance of the data set well with a slightly higher spread of predictions for some parameters. The slightly higher spread can be observed visually in fig 4.22 when compared to the same plot for better performing models in terms of MAE and EVS.

The more overall stable models, random forest and XGBoost, perform almost on the same level as the SVR model in terms of R^2- and EVS score, but with a slight lower MAE and MSE. The two models when compared to each other are almost exactly the same, both in terms of overall and parameter specific performance. This can also be confirmed by observing fig 4.7 and fig 4.12, where the plot of predicted vs actual value showcases the same patterns for both models across parameters. The random forest and XGBoost models can be considered to have better results than the SVR model in terms of variance in predictions. Comparing LLE (fig 4.18 & fig 4.8 prediction results between XGBoost and SVR it can be concluded that the SVR model has a higher maximum score, but also a lower minimum score.

The NN model when looked at in on overall sense is performing lower than the rest in terms of R^2 score, but better in terms of MAE and MSE. The low values of MAE and MSE explained in the overall sense is not a very accurate depiction of its true performance and when the specific parameters are observed individually the true gaps in the model can be found. The NN model compared to other models achieve the highest R^2 score for pitch parameters (fig 4.15), lower LLE radial sections (fig 4.13) and middle radial sections of LTE (fig 4.14). Despite the very high prediction

scores for these individual parameters the NN model has a good amount of pitfalls where performance goes below an R^2 score of 0. This can be observed in fig 4.16 and fig 4.14 and further visualized in fig 4.17 where some predictions highly diverts from the rest. A possible reason for why the model make these predictions is overfitting the training data, resulting in the model not being able to generalize data well. In summary the NN model has some outstanding qualities in predicting certain parameters or radial sections of parameters, but the results have a very high variance leading to a moderate performance in comparison to other models.

A common occurrence across all models is that LLE and LTE predictions achieve a lesser R^2 score as radial sections starts to reach 1. The phenomenon can be explained by the data set where the data points closer to the end sections are not as consistent. The inconsistency towards end of the blade comes from the blade tip often being more heavily curved when compared to the rest of the blade resulting in potential length values that are difficult for the models to understand. This could also be the effect of a small data set where not all possibilities of propeller geometry is explored leading to gaps in the data set resulting in less accurate predictions.

The whole ML pipeline performs at a general good level, but with room for improvements. The better models explains about 70% of the total variance with dips in accuracy for certain parameters radial sections most likely due to inconsistencies in the data. To achieve the best possible predictions results it is believed that the approach of combining the highest accuracy predictions between models into one final result. The goal of the predictions is to generate the first geometry model iteration to be used in the design process, meaning that a perfect geometric profile is desirable but not necessary due to the detailed optimization work that follows after the geometric blade profile has been established.

5.2 Limitations of the approach

The largest limiting factor of this approach lies in the data-set itself. The data set is small, which can lead to that certain characteristics of the individual propellers does not get explained well and lead to variance in the results. If more propeller designs had been available a larger data set could have been constructed. If such a data set had been available there is a possibility that predictions would be even more accurate.

Inside the data set there were missing data points, leading to variance in the data itself potentially setting back the models prediction accuracy. Attempts to replace missing or false data points were made, but the accuracy of the replacements is difficult to evaluate. A perfect data set in terms of the values themselves is believed to have improved the models ability to understand the variance of the data set.

The choice to not involve computational power as a factor is potentially limiting. More available computational power allows for higher amounts of iterations and more complex configurations of the models. With more computational power it is uncertain that it directly would lead to more higher scoring predictions, but the predictions would be more accurate for the data available. This means that the model potentially could learn more with more processing power available, but more complex configurations could also lead to overfitting so balance is necessary.

Values for radial sections all start at the r/R position 0.3 due to the very first radial section not being consequent across propeller designs. Performing this approach reduces variation in the data due to the removal of the first radial position which is a variable dependent on hub size. This approach makes a full 3D geometry of the blade slightly less detailed since the very bottom part of it is missing. The choice of camber value only measures the distance from the chord line to the camber line right in the middle between the two edges. This is a simplification choice that was deemed sufficient, but adding in the distance along all of the chord line would result in a more accurate depiction of the propeller blades geometry.

The blade outline is also not present in the methodology of this study. This is due to defining it through connecting lines between the endpoints of LLE and LTE creates a rough outline. It is believed that using the endpoints to draw the outline in such a manner is sufficient for the purpose of the study. However, the outline can divert in certain areas and this approach has the potential to miss special characteristics of the blade outline.

5.3 Suggestions for future work

This research study is by no means perfect and there are much room for improvements, both within the same general approach and on the topic as a whole. There are many factors holding the study back in terms of prediction accuracy and in the parameters used for the blade shape. This section will present some recommenda-

tions for future work regarding this study and the topic on a broader level.

A deep dive into the used approach for this study has a potential for much better results. This is with the assumption of a larger data set being used not necessarily based on the same data set. Performing the study with a larger quantity of data opens up for a different methodology and definition of the models being used. This could provide a different point of view to the problem while staying very related to it both in terms of marine engineering, but also in terms of ML as a topic.

On the same topic as this study with the geometric data used as output can be expanded to its raw, more precise form. An example approach would be to research how large of data set and how much processing power is needed for a ML pipeline to accurately predict the full geometric data of a propeller blade. This creates an interesting study in terms of exploring the limits of ML within the field, and could be potentially helpful for further studies utilizing ML.

Exploring the topic of ML for geometric propeller design combined with the optimization of the created design. This could be seen as a way to automate the design process with minimal human input. This allows the firstly predicted geometrical model to then self-optimize making the initial model even better reducing the time cost of blade design even further.

The overall ML method have potential to be changed, moving away from supervised learning and researching the use of unsupervised and reinforcement learning methods. This opens up for the inclusion of other models and new configurations which could be beneficial in determining which method and model is the best for propeller design.

Continuing the work on these suggested topics could help in further explain the results that was achieved in this study while also providing options as for how to further improve this study's methodology.

6

Conclusion

This has been a study on the implementation of a data driven ML model for propeller design. It has focused deeply on how the use of supervised regression models can be used to predict the first initial geometric blade shape that is created in the propeller design process. The study aimed to showcase how usable such an approach is to the propeller design field with the intent to reduce costly processes when designing the very first iteration of a blade design.

A big challenge in the study was on how to best represent the geometric blade data in a ML context. The raw geometric data of a propeller blade consisted of a large amount of data points describing the shape and form of the blade in three dimensional space. For this data to be usable it had to be transformed into a more digestible format, making certain simplifications for the sake of less complexity. The result of such simplifications and feature engineering is a simple geometric model with the necessary information to design a fully fledged propeller blade, but with less data points. The created data set was used by a ML pipeline consisting of four models making predictions on the data. The used models where: Random forest, XGBoost, NN and SVR. The models where individually tuned and tested to reach the highest possible prediction accuracy. The extracted performance metric results provided insight into the behavior of the models, describing what was easier and harder for the models to predict. The models when put together to make a final prediction was able to explain 70% of the variance of the data set. This is a result that is far from perfect but is believed to be on the better end. Potential reasons that hold the models back lie in the structure and quantity of data available. The data had certain missing values and the amount of available propeller was on the lower end, reducing the quantity of available data.

The results of this study leaves room for future research to improve both the data processing and ML part of the methodology. The results produced are believed to be beneficial for the topic of propeller design in different aspects. The reduction in resource- and time costs are viewed in a positive way, but the ethical question over replacing what was normally human work with automated ML models can be seen as negative from certain point of views. Overall the advancement showcased by this study is a strong indicator for the usefulness of ML models within the field.

7

References

Abbasi, M. M., Asif, A. M., Kleinsorge, L., & Kistner, G. (2024). Analysis of the damping and added mass properties of the marine propeller. In *8th international symposium on marine propulsors.* doi: 10.15480/882.9297

Awad, M., Khanna, R., Awad, M., & Khanna, R. (2015). Support vector regression. *Efficient learning machines: Theories, concepts, and applications for engineers and system designers*, 67–80.

Bašić, B. B., Krčum, M., & Jurić, Z. (2024, 5). Propeller optimization in marine power systems: Exploring its contribution and correlation with renewable energy solutions. *Journal of Marine Science and Engineering*, *12*. doi: 10.3390/JMSE12050843

Benini, E. (2004, 6). Significance of blade element theory in performance prediction of marine propellers. *Ocean Engineering*, *31*, 957-974. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0029801804000022` doi: 10.1016/J.OCEANENG.2003.12.001

Bertram, V. (2012). Chapter 2 - propellers. In V. Bertram (Ed.), *Practical ship hydrodynamics (second edition)* (Second Edition ed., p. 41-72). Oxford: Butterworth-Heinemann. Retrieved from `https://www.sciencedirect.com/science/article/pii/B9780080971506100028` doi: https://doi.org/10.1016/B978-0-08-097150-6.10002-8

Birk, L. (2019). *Fundamentals of ship hydrodynamics - fluid mechanics, ship resistance and propulsion.* John Wiley Sons. Retrieved from `https://app.knovel.com/hotlink/toc/id:kpFSHFMSRK/fundamentals-ship-hydrodynamics/fundamentals-ship-hydrodynamics`

Boumediene, K., Belhenniche, S., & Bouzit, M. (2018, 6). *Computational hydrodynamic analysis of a highly skewed marine propeller.* Retrieved from `http://www.preprints.org/manuscript/201806.0307/v1` doi: 10.20944/preprints201806.0307.v1

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Carlton, J. (2018). *Marine propellers and propulsion.* Butterworth-Heinemann. Retrieved from `https://books.google.se/books?id=2drWDgAAQBAJ`

Chauvin, Y., & Rumelhart, D. E. (2013). *Backpropagation: theory, architectures, and applications*. Psychology press.

Chen, T., & Guestrin, C. (2016, 1). Xgboost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-August-2016*, 785-794. doi: 10.1145/2939672.2939785

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*, 273–297.

Dietterich, T. G. (1998). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting and randomization. *Machine learning*, *32*, 1–22.

Doijode, P. S., Hickel, S., van Terwisga, T., & Visser, K. (2022). A machine learning approach for propeller design and optimization: Part i. *Applied Ocean Research*, *124*, 103178. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0141118722001213` doi: https://doi.org/10.1016/j.apor.2022.103178

Dong, G., & Liu, H. (2018). *Feature engineering for machine learning and data analytics*. CRC press.

Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: a methodology review. *Journal of biomedical informatics*, *35*(5-6), 352–359.

Ebrahimi, A., Seif, M. S., & Nouri-Borujerdi, A. (2019, 9). Hydro-acoustic and hydrodynamic optimization of a marine propeller using genetic algorithm, boundary element method, and fw-h equations. *Journal of Marine Science and Engineering*, *7*. doi: 10.3390/jmse7090321

Faltinsen, O. M. (2005). *Hydrodynamics of high-speed marine vehicles*. Cambridge University Press. Retrieved from `https://app.knovel.com/hotlink/toc/id:kpHHSMV002/hydrodynamics-high-speed/hydrodynamics-high-speed`

Franc, J.-P., & Michel, J.-M. (2006). *Fundamentals of cavitation* (Vol. 76). Springer science & Business media.

Gypa, I., Jansson, M., & Bensow, R. (2024, 1). Marine propeller optimisation through user interaction and machine learning for advanced blade design scenarios. *Ships and Offshore Structures*. Retrieved from `https://www.tandfonline.com/doi/abs/10.1080/17445302.2023.2265118` doi: 10.1080/17445302.2023.2265118

Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. In *Neural networks for perception* (pp. 65–93). Elsevier.

IMO. (2023). *2023 imo strategy on reduction of ghg emissions from ships.* Retrieved from `https://www.imo.org/en/OurWork/Environment/Pages/2023-IMO-Strategy-on-Reduction-of-GHG-Emissions-from-Ships.aspx`

Khose, A. (2020). Design parameters and manufacturing methods of marine propellers for increased efficiency in ship propulsion. *International Journal of Engineering Research and Applications (IJERA)*, *10*(09), 56-61. Retrieved from https://www.ijera.com/papers/vol10no9/Series-2/H1009025661.pdf doi: 10.9790/9622-1009025661

Little, M. A. (2019). *Machine learning for signal processing: Data science, algorithms, and computational statistics*. Oxford University Press, Incorporated.

Liu, Y., Wang, Y., & Zhang, J. (2012). New machine learning algorithm: Random forest. In *Information computing and applications (icica 2012)* (Vol. 7473, pp. 246–252). Springer. Retrieved from https://doi.org/10.1007/978-3-642-34062-8_32 doi: 10.1007/978-3-642-34062-8_32

López, O. A. M., López, A. M., & Crossa, J. (2022). Support vector machines and support vector regression. In *Multivariate statistical machine learning methods for genomic prediction* (pp. 337–378). Springer.

Mitchell, T. (1997). *Machine learning* (Vol. 1) (No. 9). McGraw-hill New York.

Moon, B.-Y., Sun, M.-Y., & Lee, K.-Y. (2014). Study on flow analysis of three-dimensional screw propeller with respect to rotational speed variable. *Original Research Article Journal of Ocean Engineering and Technology*, *28*, 500-507. Retrieved from http://dx. doi: 10.5574/KSOE.2014.28.6.500

Naqa, I. E., & Murphy, M. J. (2015). What is machine learning? *Machine Learning in Radiation Oncology*, 3-11. Retrieved from https://link.springer.com/chapter/10.1007/978-3-319-18305-3_1 doi: 10.1007/978-3-319-18305-3_1

Nielsen, D. (2016). *Tree boosting with xgboost-why does xgboost win" every" machine learning competition?* (Unpublished master's thesis). NTNU.

Nwanganga, C. M., Fred. (2020). *Practical machine learning in r*. John Wiley Sons. Retrieved from https://app.knovel.com/hotlink/toc/id:kpPMLR0005/practical-machine-learning/practical-machine-learning

Pan, Z., Sun, X. L., Zhang, Y., Li, X. B., Zhu, X. K., & Yang, L. C. (2023). Study on marine propeller design using artificial neural networks and multivariate analysis. *2023 IEEE 11th International Conference on Computer Science and Network Technology, ICCSNT 2023*, 169-180. doi: 10.1109/ICCSNT58790.2023.10334601

Quinlan, J. R. (1986, 1). Induction of decision trees. *Machine Learning*, *1*, 81-106. doi: 10.1007/BF00116251

Radosavljevic, Z., Whitworth. (2011). *7. stern challenge - getting it right.* The Royal Institution of Naval Architects (RINA). Retrieved from https://app.knovel.com/hotlink/khtml/id:kt00BHQ0K1/marine-cfd-2011/stern-challenge-getting

Rafiq, M. Y., Bugmann, G., & Easterbrook, D. J. (2001, 1). Neural network design for engineering applications. *Computers Structures*, *79*, 1541-1552. doi: 10.1016/S0045-7949(01)00039-6

Riccardo, T., & Flavio, C. (2021). Machine learning regression techniques for the modeling of complex systems: An overview. *IEEE Electromagnetic Compatibility Magazine*, *10*(4), 71-79. doi: 10.1109/MEMC.2021.9705310

Rong, S., & Bao-wen, Z. (2018). The research of regression model in machine learning field. *MATEC Web Conf.*, *176*, 01033. Retrieved from `https://doi.org/10.1051/matecconf/201817601033` doi: 10.1051/matecconf/201817601033

Sadiq, M. S., Loya, A., Mushtaque, I., & Akram, W. (2025, Mar.). Effect of rake angle on dtmb marine propeller. *Sustainable Marine Structures*, *7*(1), 21–34. Retrieved from `https://journals.nasspublishing.com/index.php/sms/article/view/1650` doi: 10.36956/sms.v7i1.1650

Smilkov, D., Thorat, N., Kim, B., Viégas, F., & Wattenberg, M. (2025). *Tensorflow playground.* Retrieved from `https://playground.tensorflow.org` (Interactive visualization)

Smola, A. J., & Schölkopf, B. (2004, 8). A tutorial on support vector regression. *Statistics and Computing*, *14*, 199-222. doi: 10.1023/B:STCO.0000035301.49549.88

Tadros, M., Shi, W., Xu, Y., & Song, Y. (2024a). A unified cross-series marine propeller design method based on machine learning. *Ocean Engineering*, *314*, 119691. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0029801824030294` doi: https://doi.org/10.1016/j.oceaneng.2024.119691

Tadros, M., Shi, W., Xu, Y., & Song, Y. (2024b, 1). A unified cross-series marine propeller design method based on machine learning. *Ocean Engineering*, *314*. doi: 10.1016/J.OCEANENG.2024.119691

Wang, H. (2016). Machine learning basics. *Deep learning*, 98–164.

Zhang, F., & O'Donnell, L. J. (2020). Chapter 7 - support vector regression. In A. Mechelli & S. Vieira (Eds.), *Machine learning* (p. 123-140). Academic Press. Retrieved from `https://www.sciencedirect.com/science/article/pii/B9780128157398000079` doi: https://doi.org/10.1016/B978-0-12-815739-8.00007-9

Zhou, Z.-H. (2021). *Machine learning.* Springer nature.

Zou, J., Han, Y., & So, S.-S. (2009). Overview of artificial neural networks. *Artificial neural networks: methods and applications*, 14–22.

A

Appendix 1

Detailed Outputs

Random forest

```
Overall Performance Metrics for Random Forest (Tuned):
Mean Absolute Error: 161.4846
Explained Variance Score: 0.7152
Mean Squared Error: 110278.2722
R² Score: 0.6915

Per-Output Performance Metrics for Random Forest (Tuned):
P0.3:
  R² Score: 0.5507
  Mean Absolute Error: 560.8076
  Mean Squared Error: 451056.4769
  Explained Variance Score: 0.6590
P0.35:
  R² Score: 0.5719
  Mean Absolute Error: 546.6574
  Mean Squared Error: 439825.1510
  Explained Variance Score: 0.6741
P0.4:
  R² Score: 0.6043
  Mean Absolute Error: 525.6176
  Mean Squared Error: 408437.9784
  Explained Variance Score: 0.7038
P0.5:
  R² Score: 0.6507
  Mean Absolute Error: 504.4113
  Mean Squared Error: 391766.6465
  Explained Variance Score: 0.7328
P0.6:
  R² Score: 0.6883
  Mean Absolute Error: 480.7834
```

```
  Mean Squared Error: 380220.6447
  Explained Variance Score: 0.7561
P0.7:
  R² Score: 0.7120
  Mean Absolute Error: 463.3468
  Mean Squared Error: 375198.4230
  Explained Variance Score: 0.7707
P0.8:
  R² Score: 0.7078
  Mean Absolute Error: 466.2749
  Mean Squared Error: 397600.5071
  Explained Variance Score: 0.7526
P0.9:
  R² Score: 0.6929
  Mean Absolute Error: 486.5763
  Mean Squared Error: 415972.3777
  Explained Variance Score: 0.7346
P0.95:
  R² Score: 0.6714
  Mean Absolute Error: 504.2862
  Mean Squared Error: 435991.8160
  Explained Variance Score: 0.7102
P0.975:
  R² Score: 0.6656
  Mean Absolute Error: 505.4987
  Mean Squared Error: 440227.1863
  Explained Variance Score: 0.6965
P1:
  R² Score: 0.6505
  Mean Absolute Error: 519.0295
  Mean Squared Error: 456855.0536
  Explained Variance Score: 0.6787
LE0.3:
  R² Score: 0.7938
  Mean Absolute Error: 32.6160
  Mean Squared Error: 2631.1933
  Explained Variance Score: 0.8191
LE0.35:
  R² Score: 0.8061
  Mean Absolute Error: 40.9057
  Mean Squared Error: 3718.8676
  Explained Variance Score: 0.8172
LE0.4:
  R² Score: 0.8195
  Mean Absolute Error: 47.4008
  Mean Squared Error: 4381.1720
```

```
  Explained Variance Score: 0.8261
LE0.5:
  R² Score: 0.8274
  Mean Absolute Error: 57.7889
  Mean Squared Error: 6323.7785
  Explained Variance Score: 0.8298
LE0.6:
  R² Score: 0.8204
  Mean Absolute Error: 66.5096
  Mean Squared Error: 7885.9154
  Explained Variance Score: 0.8225
LE0.7:
  R² Score: 0.7946
  Mean Absolute Error: 70.4713
  Mean Squared Error: 8974.9589
  Explained Variance Score: 0.7983
LE0.8:
  R² Score: 0.7217
  Mean Absolute Error: 75.0792
  Mean Squared Error: 10110.2457
  Explained Variance Score: 0.7296
LE0.9:
  R² Score: 0.5871
  Mean Absolute Error: 82.6808
  Mean Squared Error: 9363.3488
  Explained Variance Score: 0.6073
LE0.95:
  R² Score: 0.3943
  Mean Absolute Error: 85.7789
  Mean Squared Error: 13966.2818
  Explained Variance Score: 0.3974
LE0.975:
  R² Score: 0.3298
  Mean Absolute Error: 105.5890
  Mean Squared Error: 22787.0647
  Explained Variance Score: 0.3324
LE1:
  R² Score: 0.2898
  Mean Absolute Error: 191.6653
  Mean Squared Error: 75416.5529
  Explained Variance Score: 0.2926
TE0.3:
  R² Score: 0.7946
  Mean Absolute Error: 30.9970
  Mean Squared Error: 1910.7276
  Explained Variance Score: 0.7969
```

```
TE0.35:
  R² Score: 0.7931
  Mean Absolute Error: 32.6717
  Mean Squared Error: 2262.1389
  Explained Variance Score: 0.7965
TE0.4:
  R² Score: 0.7953
  Mean Absolute Error: 34.2163
  Mean Squared Error: 2554.7674
  Explained Variance Score: 0.8002
TE0.5:
  R² Score: 0.8083
  Mean Absolute Error: 39.1502
  Mean Squared Error: 3134.7650
  Explained Variance Score: 0.8112
TE0.6:
  R² Score: 0.8158
  Mean Absolute Error: 46.7077
  Mean Squared Error: 4033.5275
  Explained Variance Score: 0.8180
TE0.7:
  R² Score: 0.8340
  Mean Absolute Error: 51.8908
  Mean Squared Error: 4964.0100
  Explained Variance Score: 0.8372
TE0.8:
  R² Score: 0.8318
  Mean Absolute Error: 64.3099
  Mean Squared Error: 6834.7199
  Explained Variance Score: 0.8394
TE0.9:
  R² Score: 0.7952
  Mean Absolute Error: 81.1760
  Mean Squared Error: 11229.4839
  Explained Variance Score: 0.8083
TE0.95:
  R² Score: 0.7558
  Mean Absolute Error: 86.4540
  Mean Squared Error: 14422.7502
  Explained Variance Score: 0.7712
TE0.975:
  R² Score: 0.6731
  Mean Absolute Error: 97.5097
  Mean Squared Error: 20277.0636
  Explained Variance Score: 0.6972
TE1:
```

```
  R² Score: 0.5874
  Mean Absolute Error: 86.4823
  Mean Squared Error: 21724.9559
  Explained Variance Score: 0.6227
C0.3:
  R² Score: 0.6536
  Mean Absolute Error: 2.8694
  Mean Squared Error: 13.1553
  Explained Variance Score: 0.6642
C0.35:
  R² Score: 0.7224
  Mean Absolute Error: 3.8233
  Mean Squared Error: 22.3754
  Explained Variance Score: 0.7225
C0.4:
  R² Score: 0.7834
  Mean Absolute Error: 3.7483
  Mean Squared Error: 24.5608
  Explained Variance Score: 0.7835
C0.5:
  R² Score: 0.7764
  Mean Absolute Error: 3.5483
  Mean Squared Error: 20.7189
  Explained Variance Score: 0.7805
C0.6:
  R² Score: 0.6344
  Mean Absolute Error: 4.0116
  Mean Squared Error: 26.3741
  Explained Variance Score: 0.6523
C0.7:
  R² Score: 0.6823
  Mean Absolute Error: 3.4046
  Mean Squared Error: 20.5708
  Explained Variance Score: 0.6958
C0.8:
  R² Score: 0.7697
  Mean Absolute Error: 2.4486
  Mean Squared Error: 13.3380
  Explained Variance Score: 0.7720
C0.9:
  R² Score: 0.7888
  Mean Absolute Error: 2.5389
  Mean Squared Error: 10.5856
  Explained Variance Score: 0.7889
C0.95:
  R² Score: 0.6759
```

```
  Mean Absolute Error: 3.1402
  Mean Squared Error: 14.9807
  Explained Variance Score: 0.6793
C0.975:
  R² Score: 0.5671
  Mean Absolute Error: 3.1843
  Mean Squared Error: 14.5807
  Explained Variance Score: 0.5919
C1:
  R² Score: 0.5388
  Mean Absolute Error: 1.2617
  Mean Squared Error: 2.1843
  Explained Variance Score: 0.5991
```

A.0.1 XGBoost

```
Overall Performance Metrics for XGBoost:
Mean Absolute Error: 160.1692
Explained Variance Score: 0.7195
Mean Squared Error: 113606.1632
R² Score: 0.6940

Per-Output Performance Metrics for XGBoost:
P0.3:
  R² Score: 0.5537
  Mean Absolute Error: 542.6838
  Mean Squared Error: 448011.9916
  Explained Variance Score: 0.6872
P0.35:
  R² Score: 0.5843
  Mean Absolute Error: 521.2520
  Mean Squared Error: 427164.1028
  Explained Variance Score: 0.6856
P0.4:
  R² Score: 0.5983
  Mean Absolute Error: 522.0202
  Mean Squared Error: 414630.0993
  Explained Variance Score: 0.6936
P0.5:
  R² Score: 0.6431
  Mean Absolute Error: 489.1666
  Mean Squared Error: 400314.5331
  Explained Variance Score: 0.7144
P0.6:
  R² Score: 0.6857
  Mean Absolute Error: 463.3885
```

```
  Mean Squared Error: 383377.1736
  Explained Variance Score: 0.7457
P0.7:
  R² Score: 0.7030
  Mean Absolute Error: 442.6612
  Mean Squared Error: 386949.9090
  Explained Variance Score: 0.7650
P0.8:
  R² Score: 0.7008
  Mean Absolute Error: 461.0577
  Mean Squared Error: 407067.9038
  Explained Variance Score: 0.7638
P0.9:
  R² Score: 0.6820
  Mean Absolute Error: 505.7311
  Mean Squared Error: 430726.4052
  Explained Variance Score: 0.7138
P0.95:
  R² Score: 0.6720
  Mean Absolute Error: 503.7566
  Mean Squared Error: 435183.7358
  Explained Variance Score: 0.7053
P0.975:
  R² Score: 0.6304
  Mean Absolute Error: 527.2257
  Mean Squared Error: 486613.8387
  Explained Variance Score: 0.6559
P1:
  R² Score: 0.6005
  Mean Absolute Error: 543.4438
  Mean Squared Error: 522258.1219
  Explained Variance Score: 0.6258
LE0.3:
  R² Score: 0.8082
  Mean Absolute Error: 28.9787
  Mean Squared Error: 2446.8603
  Explained Variance Score: 0.8371
LE0.35:
  R² Score: 0.8124
  Mean Absolute Error: 38.4831
  Mean Squared Error: 3598.4696
  Explained Variance Score: 0.8294
LE0.4:
  R² Score: 0.8171
  Mean Absolute Error: 47.1804
  Mean Squared Error: 4438.6351
```

```
  Explained Variance Score: 0.8244
LE0.5:
  R² Score: 0.8155
  Mean Absolute Error: 59.0322
  Mean Squared Error: 6759.1880
  Explained Variance Score: 0.8194
LE0.6:
  R² Score: 0.8042
  Mean Absolute Error: 67.5113
  Mean Squared Error: 8598.8409
  Explained Variance Score: 0.8076
LE0.7:
  R² Score: 0.8206
  Mean Absolute Error: 62.4806
  Mean Squared Error: 7838.6511
  Explained Variance Score: 0.8236
LE0.8:
  R² Score: 0.7788
  Mean Absolute Error: 62.2236
  Mean Squared Error: 8038.2874
  Explained Variance Score: 0.7844
LE0.9:
  R² Score: 0.6550
  Mean Absolute Error: 71.7032
  Mean Squared Error: 7824.6713
  Explained Variance Score: 0.6648
LE0.95:
  R² Score: 0.4097
  Mean Absolute Error: 79.6236
  Mean Squared Error: 13610.1646
  Explained Variance Score: 0.4180
LE0.975:
  R² Score: 0.2907
  Mean Absolute Error: 110.8128
  Mean Squared Error: 24116.8601
  Explained Variance Score: 0.3026
LE1:
  R² Score: 0.2982
  Mean Absolute Error: 191.9785
  Mean Squared Error: 74520.6965
  Explained Variance Score: 0.3048
TE0.3:
  R² Score: 0.8118
  Mean Absolute Error: 30.0728
  Mean Squared Error: 1750.8286
  Explained Variance Score: 0.8154
```

```
TE0.35:
  R² Score: 0.8222
  Mean Absolute Error: 31.2211
  Mean Squared Error: 1943.7578
  Explained Variance Score: 0.8264
TE0.4:
  R² Score: 0.8362
  Mean Absolute Error: 31.6710
  Mean Squared Error: 2044.1120
  Explained Variance Score: 0.8413
TE0.5:
  R² Score: 0.8407
  Mean Absolute Error: 38.5163
  Mean Squared Error: 2605.6906
  Explained Variance Score: 0.8429
TE0.6:
  R² Score: 0.8601
  Mean Absolute Error: 43.4431
  Mean Squared Error: 3063.4907
  Explained Variance Score: 0.8608
TE0.7:
  R² Score: 0.8678
  Mean Absolute Error: 49.2341
  Mean Squared Error: 3951.1762
  Explained Variance Score: 0.8692
TE0.8:
  R² Score: 0.8416
  Mean Absolute Error: 62.7739
  Mean Squared Error: 6438.5872
  Explained Variance Score: 0.8489
TE0.9:
  R² Score: 0.7860
  Mean Absolute Error: 83.3796
  Mean Squared Error: 11734.5690
  Explained Variance Score: 0.8000
TE0.95:
  R² Score: 0.7340
  Mean Absolute Error: 99.7961
  Mean Squared Error: 15710.6925
  Explained Variance Score: 0.7475
TE0.975:
  R² Score: 0.6698
  Mean Absolute Error: 107.8808
  Mean Squared Error: 20480.2866
  Explained Variance Score: 0.6838
TE1:
```

```
  R² Score: 0.5314
  Mean Absolute Error: 92.8995
  Mean Squared Error: 24669.6276
  Explained Variance Score: 0.5727
C0.3:
  R² Score: 0.6527
  Mean Absolute Error: 3.0550
  Mean Squared Error: 13.1876
  Explained Variance Score: 0.6662
C0.35:
  R² Score: 0.7228
  Mean Absolute Error: 3.9597
  Mean Squared Error: 22.3509
  Explained Variance Score: 0.7230
C0.4:
  R² Score: 0.7616
  Mean Absolute Error: 3.7740
  Mean Squared Error: 27.0376
  Explained Variance Score: 0.7616
C0.5:
  R² Score: 0.7080
  Mean Absolute Error: 4.2421
  Mean Squared Error: 27.0667
  Explained Variance Score: 0.7113
C0.6:
  R² Score: 0.5765
  Mean Absolute Error: 4.3074
  Mean Squared Error: 30.5528
  Explained Variance Score: 0.5939
C0.7:
  R² Score: 0.6594
  Mean Absolute Error: 3.5863
  Mean Squared Error: 22.0567
  Explained Variance Score: 0.6691
C0.8:
  R² Score: 0.7874
  Mean Absolute Error: 2.5210
  Mean Squared Error: 12.3112
  Explained Variance Score: 0.7897
C0.9:
  R² Score: 0.8105
  Mean Absolute Error: 2.3069
  Mean Squared Error: 9.4967
  Explained Variance Score: 0.8106
C0.95:
  R² Score: 0.7297
```

```
  Mean Absolute Error: 2.6892
  Mean Squared Error: 12.4937
  Explained Variance Score: 0.7355
C0.975:
  R² Score: 0.6996
  Mean Absolute Error: 2.5054
  Mean Squared Error: 10.1187
  Explained Variance Score: 0.7331
C1:
  R² Score: 0.4613
  Mean Absolute Error: 1.2142
  Mean Squared Error: 2.5513
  Explained Variance Score: 0.5851
```

A.0.2 Neural Networks

```
Overall Performance Metrics for Neural Network:
Mean Absolute Error: 145.0363
Explained Variance Score: 0.6179
Mean Squared Error: 83107.8334
R² Score: 0.5734

Per-Output Performance Metrics for Neural Network:
P0.3:
  R² Score: 0.6127
  Mean Absolute Error: 515.4364
  Mean Squared Error: 388775.3686
  Explained Variance Score: 0.6590
P0.35:
  R² Score: 0.6484
  Mean Absolute Error: 497.3385
  Mean Squared Error: 361293.1250
  Explained Variance Score: 0.6933
P0.4:
  R² Score: 0.7582
  Mean Absolute Error: 439.5723
  Mean Squared Error: 249543.1200
  Explained Variance Score: 0.7836
P0.5:
  R² Score: 0.8322
  Mean Absolute Error: 364.2744
  Mean Squared Error: 188232.8747
  Explained Variance Score: 0.8515
P0.6:
  R² Score: 0.8674
  Mean Absolute Error: 315.1589
```

```
  Mean Squared Error: 161770.1789
  Explained Variance Score: 0.8814
P0.7:
  R² Score: 0.8778
  Mean Absolute Error: 299.9779
  Mean Squared Error: 159183.0620
  Explained Variance Score: 0.8932
P0.8:
  R² Score: 0.8727
  Mean Absolute Error: 319.8876
  Mean Squared Error: 173273.8747
  Explained Variance Score: 0.8885
P0.9:
  R² Score: 0.7894
  Mean Absolute Error: 415.1634
  Mean Squared Error: 285217.4237
  Explained Variance Score: 0.8221
P0.95:
  R² Score: 0.6766
  Mean Absolute Error: 454.7310
  Mean Squared Error: 429116.3637
  Explained Variance Score: 0.7342
P0.975:
  R² Score: 0.7271
  Mean Absolute Error: 452.8788
  Mean Squared Error: 359288.2850
  Explained Variance Score: 0.7715
P1:
  R² Score: 0.6401
  Mean Absolute Error: 503.1422
  Mean Squared Error: 470439.6824
  Explained Variance Score: 0.6857
LE0.3:
  R² Score: 0.9056
  Mean Absolute Error: 28.7739
  Mean Squared Error: 1204.9642
  Explained Variance Score: 0.9085
LE0.35:
  R² Score: 0.8940
  Mean Absolute Error: 38.1891
  Mean Squared Error: 2032.0673
  Explained Variance Score: 0.8942
LE0.4:
  R² Score: 0.8484
  Mean Absolute Error: 47.9401
  Mean Squared Error: 3679.8134
```

```
  Explained Variance Score: 0.8503
LE0.5:
  R² Score: 0.8358
  Mean Absolute Error: 62.5085
  Mean Squared Error: 6017.0601
  Explained Variance Score: 0.8365
LE0.6:
  R² Score: 0.8522
  Mean Absolute Error: 66.7484
  Mean Squared Error: 6491.2097
  Explained Variance Score: 0.8538
LE0.7:
  R² Score: 0.8063
  Mean Absolute Error: 70.4869
  Mean Squared Error: 8461.6370
  Explained Variance Score: 0.8148
LE0.8:
  R² Score: 0.4905
  Mean Absolute Error: 99.2558
  Mean Squared Error: 18513.3018
  Explained Variance Score: 0.5641
LE0.9:
  R² Score: 0.5125
  Mean Absolute Error: 86.3608
  Mean Squared Error: 11057.1154
  Explained Variance Score: 0.6260
LE0.95:
  R² Score: 0.3405
  Mean Absolute Error: 78.1188
  Mean Squared Error: 15205.4411
  Explained Variance Score: 0.3649
LE0.975:
  R² Score: 0.1407
  Mean Absolute Error: 130.5271
  Mean Squared Error: 29216.3550
  Explained Variance Score: 0.2433
LE1:
  R² Score: 0.4428
  Mean Absolute Error: 157.9989
  Mean Squared Error: 59165.6426
  Explained Variance Score: 0.4462
TE0.3:
  R² Score: 0.8424
  Mean Absolute Error: 29.5101
  Mean Squared Error: 1465.9766
  Explained Variance Score: 0.8519
```

```
TE0.35:
  R² Score: 0.8906
  Mean Absolute Error: 24.9098
  Mean Squared Error: 1196.2226
  Explained Variance Score: 0.8975
TE0.4:
  R² Score: 0.9033
  Mean Absolute Error: 26.4440
  Mean Squared Error: 1206.6273
  Explained Variance Score: 0.9081
TE0.5:
  R² Score: 0.9442
  Mean Absolute Error: 20.6242
  Mean Squared Error: 912.1295
  Explained Variance Score: 0.9458
TE0.6:
  R² Score: 0.9233
  Mean Absolute Error: 28.3466
  Mean Squared Error: 1679.2127
  Explained Variance Score: 0.9234
TE0.7:
  R² Score: 0.8943
  Mean Absolute Error: 45.3996
  Mean Squared Error: 3160.7982
  Explained Variance Score: 0.8961
TE0.8:
  R² Score: 0.8633
  Mean Absolute Error: 63.3766
  Mean Squared Error: 5555.0737
  Explained Variance Score: 0.8638
TE0.9:
  R² Score: 0.7126
  Mean Absolute Error: 94.6215
  Mean Squared Error: 15757.7040
  Explained Variance Score: 0.7259
TE0.95:
  R² Score: 0.5918
  Mean Absolute Error: 113.9828
  Mean Squared Error: 24115.6313
  Explained Variance Score: 0.6179
TE0.975:
  R² Score: 0.3458
  Mean Absolute Error: 144.8455
  Mean Squared Error: 40573.4424
  Explained Variance Score: 0.4212
TE1:
```

```
  R² Score: -2.2975
  Mean Absolute Error: 299.1308
  Mean Squared Error: 173615.3269
  Explained Variance Score: -1.5468
C0.3:
  R² Score: -0.0413
  Mean Absolute Error: 4.9496
  Mean Squared Error: 39.5410
  Explained Variance Score: 0.0986
C0.35:
  R² Score: 0.6200
  Mean Absolute Error: 4.1622
  Mean Squared Error: 30.6345
  Explained Variance Score: 0.6329
C0.4:
  R² Score: 0.6962
  Mean Absolute Error: 4.9386
  Mean Squared Error: 34.4502
  Explained Variance Score: 0.7167
C0.5:
  R² Score: 0.7317
  Mean Absolute Error: 3.9460
  Mean Squared Error: 24.8691
  Explained Variance Score: 0.7460
C0.6:
  R² Score: 0.6657
  Mean Absolute Error: 3.8660
  Mean Squared Error: 24.1224
  Explained Variance Score: 0.6704
C0.7:
  R² Score: 0.5678
  Mean Absolute Error: 3.9554
  Mean Squared Error: 27.9834
  Explained Variance Score: 0.6422
C0.8:
  R² Score: 0.5719
  Mean Absolute Error: 3.7109
  Mean Squared Error: 24.7945
  Explained Variance Score: 0.6147
C0.9:
  R² Score: 0.4294
  Mean Absolute Error: 4.2342
  Mean Squared Error: 28.6031
  Explained Variance Score: 0.4391
C0.95:
  R² Score: -0.0565
```

```
  Mean Absolute Error: 5.5936
  Mean Squared Error: 48.8323
  Explained Variance Score: -0.0232
C0.975:
  R² Score: -0.2272
  Mean Absolute Error: 5.4559
  Mean Squared Error: 41.3388
  Explained Variance Score: -0.2041
C1:
  R² Score: 0.2847
  Mean Absolute Error: 1.1232
  Mean Squared Error: 3.3873
  Explained Variance Score: 0.2848
```

A.0.3 SVR

```
Overall Performance Metrics for Support Vector Regression:
Mean Absolute Error: 166.5979
Explained Variance Score: 0.7186
Mean Squared Error: 121498.0156
R² Score: 0.7031

Per-Output Performance Metrics for Support Vector Regression:
P0.3:
  R² Score: 0.5112
  Mean Absolute Error: 578.9993
  Mean Squared Error: 490694.5558
  Explained Variance Score: 0.5386
P0.35:
  R² Score: 0.5358
  Mean Absolute Error: 567.8633
  Mean Squared Error: 476934.6861
  Explained Variance Score: 0.5733
P0.4:
  R² Score: 0.5668
  Mean Absolute Error: 546.9758
  Mean Squared Error: 447157.3517
  Explained Variance Score: 0.6122
P0.5:
  R² Score: 0.6188
  Mean Absolute Error: 519.3693
  Mean Squared Error: 427619.6407
  Explained Variance Score: 0.6684
P0.6:
  R² Score: 0.6668
  Mean Absolute Error: 486.8545
```

```
  Mean Squared Error: 406525.7278
  Explained Variance Score: 0.7074
P0.7:
  R² Score: 0.6882
  Mean Absolute Error: 466.5360
  Mean Squared Error: 406242.1660
  Explained Variance Score: 0.7095
P0.8:
  R² Score: 0.6750
  Mean Absolute Error: 494.9893
  Mean Squared Error: 442205.9745
  Explained Variance Score: 0.6771
P0.9:
  R² Score: 0.6671
  Mean Absolute Error: 510.6964
  Mean Squared Error: 450928.1336
  Explained Variance Score: 0.6729
P0.95:
  R² Score: 0.6223
  Mean Absolute Error: 570.8208
  Mean Squared Error: 501194.5806
  Explained Variance Score: 0.6223
P0.975:
  R² Score: 0.5850
  Mean Absolute Error: 603.4797
  Mean Squared Error: 546358.1136
  Explained Variance Score: 0.5851
P1:
  R² Score: 0.5999
  Mean Absolute Error: 596.5210
  Mean Squared Error: 523096.8132
  Explained Variance Score: 0.5999
LE0.3:
  R² Score: 0.8927
  Mean Absolute Error: 23.6183
  Mean Squared Error: 1368.5996
  Explained Variance Score: 0.8951
LE0.35:
  R² Score: 0.9009
  Mean Absolute Error: 27.7106
  Mean Squared Error: 1899.8744
  Explained Variance Score: 0.9014
LE0.4:
  R² Score: 0.9060
  Mean Absolute Error: 29.2533
  Mean Squared Error: 2281.4348
```

```
  Explained Variance Score: 0.9061
LE0.5:
  R² Score: 0.8984
  Mean Absolute Error: 39.0192
  Mean Squared Error: 3723.4615
  Explained Variance Score: 0.9008
LE0.6:
  R² Score: 0.8864
  Mean Absolute Error: 45.3245
  Mean Squared Error: 4988.3484
  Explained Variance Score: 0.8884
LE0.7:
  R² Score: 0.8655
  Mean Absolute Error: 49.6582
  Mean Squared Error: 5877.5184
  Explained Variance Score: 0.8664
LE0.8:
  R² Score: 0.8186
  Mean Absolute Error: 51.2277
  Mean Squared Error: 6589.9466
  Explained Variance Score: 0.8192
LE0.9:
  R² Score: 0.6320
  Mean Absolute Error: 75.2591
  Mean Squared Error: 8345.8377
  Explained Variance Score: 0.6705
LE0.95:
  R² Score: 0.3885
  Mean Absolute Error: 105.2498
  Mean Squared Error: 14098.6590
  Explained Variance Score: 0.4645
LE0.975:
  R² Score: 0.2452
  Mean Absolute Error: 143.2475
  Mean Squared Error: 25663.6861
  Explained Variance Score: 0.3794
LE1:
  R² Score: 0.3170
  Mean Absolute Error: 227.2785
  Mean Squared Error: 72524.2266
  Explained Variance Score: 0.3185
TE0.3:
  R² Score: 0.8960
  Mean Absolute Error: 22.4884
  Mean Squared Error: 967.9687
  Explained Variance Score: 0.8991
```

```
TE0.35:
  R² Score: 0.8815
  Mean Absolute Error: 27.7811
  Mean Squared Error: 1295.7685
  Explained Variance Score: 0.8913
TE0.4:
  R² Score: 0.8900
  Mean Absolute Error: 28.6491
  Mean Squared Error: 1372.5961
  Explained Variance Score: 0.8995
TE0.5:
  R² Score: 0.8929
  Mean Absolute Error: 32.4120
  Mean Squared Error: 1750.8105
  Explained Variance Score: 0.9065
TE0.6:
  R² Score: 0.8935
  Mean Absolute Error: 38.7301
  Mean Squared Error: 2331.5511
  Explained Variance Score: 0.9088
TE0.7:
  R² Score: 0.9234
  Mean Absolute Error: 38.5790
  Mean Squared Error: 2291.4105
  Explained Variance Score: 0.9269
TE0.8:
  R² Score: 0.9361
  Mean Absolute Error: 34.5469
  Mean Squared Error: 2599.0152
  Explained Variance Score: 0.9393
TE0.9:
  R² Score: 0.8720
  Mean Absolute Error: 58.1878
  Mean Squared Error: 7016.8617
  Explained Variance Score: 0.8721
TE0.95:
  R² Score: 0.8174
  Mean Absolute Error: 71.7883
  Mean Squared Error: 10787.3844
  Explained Variance Score: 0.8174
TE0.975:
  R² Score: 0.6955
  Mean Absolute Error: 83.9348
  Mean Squared Error: 18884.2022
  Explained Variance Score: 0.7012
TE1:
```

```
  R² Score: 0.4283
  Mean Absolute Error: 98.2141
  Mean Squared Error: 30097.8899
  Explained Variance Score: 0.4534
C0.3:
  R² Score: 0.6262
  Mean Absolute Error: 3.0346
  Mean Squared Error: 14.1932
  Explained Variance Score: 0.6460
C0.35:
  R² Score: 0.7500
  Mean Absolute Error: 3.4422
  Mean Squared Error: 20.1576
  Explained Variance Score: 0.7590
C0.4:
  R² Score: 0.8046
  Mean Absolute Error: 3.3709
  Mean Squared Error: 22.1593
  Explained Variance Score: 0.8122
C0.5:
  R² Score: 0.7241
  Mean Absolute Error: 3.8869
  Mean Squared Error: 25.5686
  Explained Variance Score: 0.7246
C0.6:
  R² Score: 0.5209
  Mean Absolute Error: 4.7736
  Mean Squared Error: 34.5679
  Explained Variance Score: 0.5250
C0.7:
  R² Score: 0.5197
  Mean Absolute Error: 4.4728
  Mean Squared Error: 31.1009
  Explained Variance Score: 0.5254
C0.8:
  R² Score: 0.6783
  Mean Absolute Error: 3.3873
  Mean Squared Error: 18.6316
  Explained Variance Score: 0.7135
C0.9:
  R² Score: 0.8465
  Mean Absolute Error: 2.3377
  Mean Squared Error: 7.6945
  Explained Variance Score: 0.8603
C0.95:
  R² Score: 0.7973
```

```
  Mean Absolute Error: 2.5020
  Mean Squared Error: 9.3696
  Explained Variance Score: 0.8021
C0.975:
  R² Score: 0.6554
  Mean Absolute Error: 2.5591
  Mean Squared Error: 11.6096
  Explained Variance Score: 0.6554
C1:
  R² Score: 0.4009
  Mean Absolute Error: 1.2746
  Mean Squared Error: 2.8372
  Explained Variance Score: 0.4009
```

A.1 Additional figures

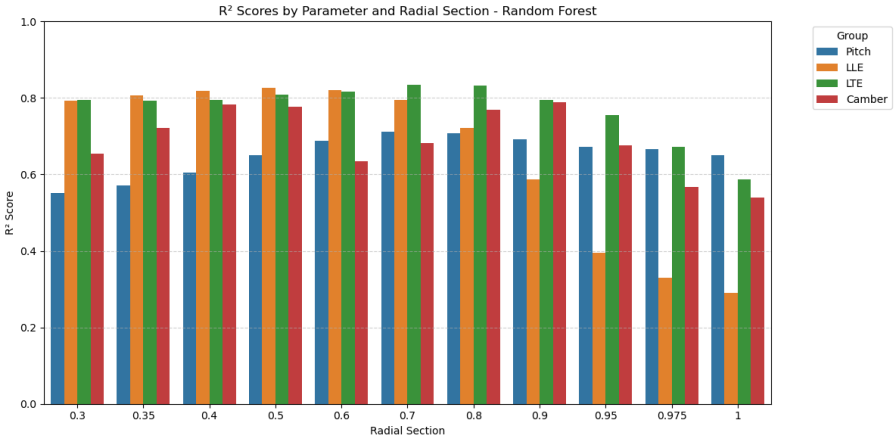


Figure A.1: All random forest outputs plotted over radial sections

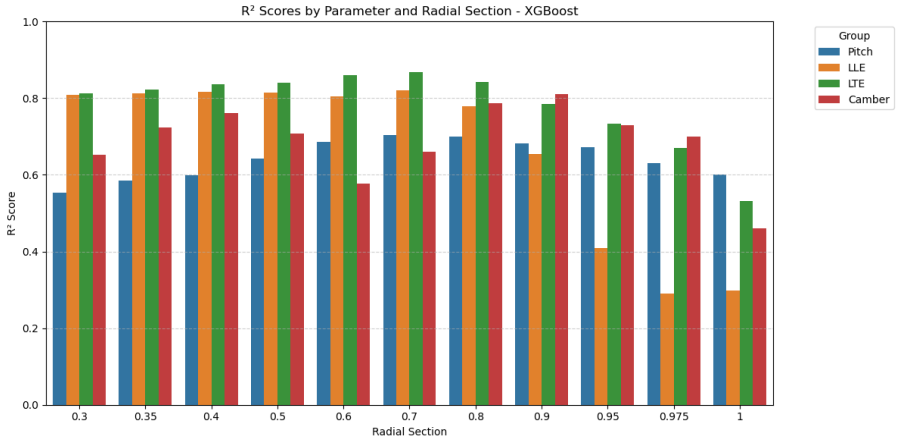


Figure A.2: All XGBoost outputs plotted over radial sections

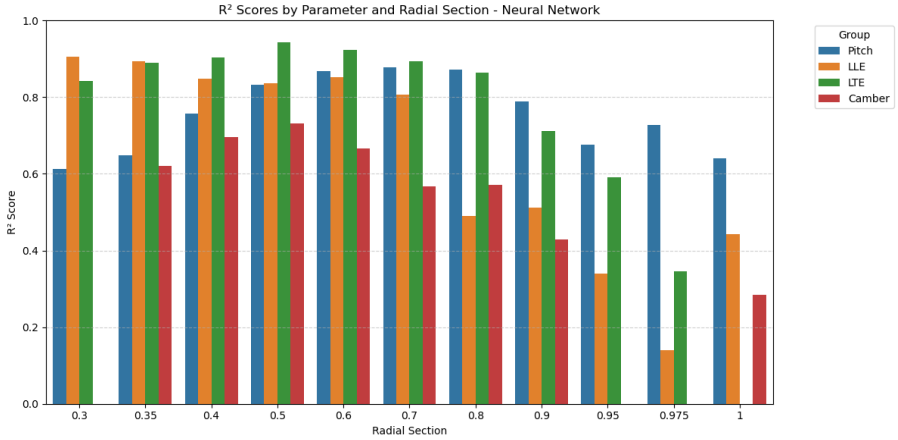


Figure A.3: All NN outputs plotted over radial sections

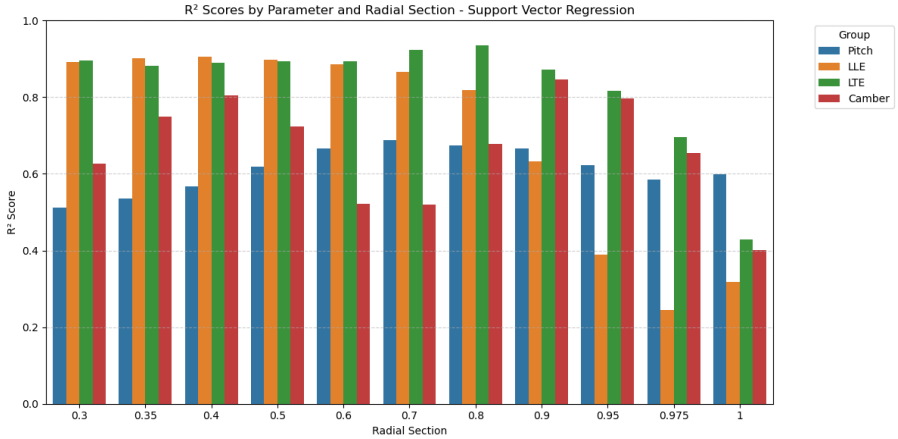


Figure A.4: All SVR outputs plotted over radial sections

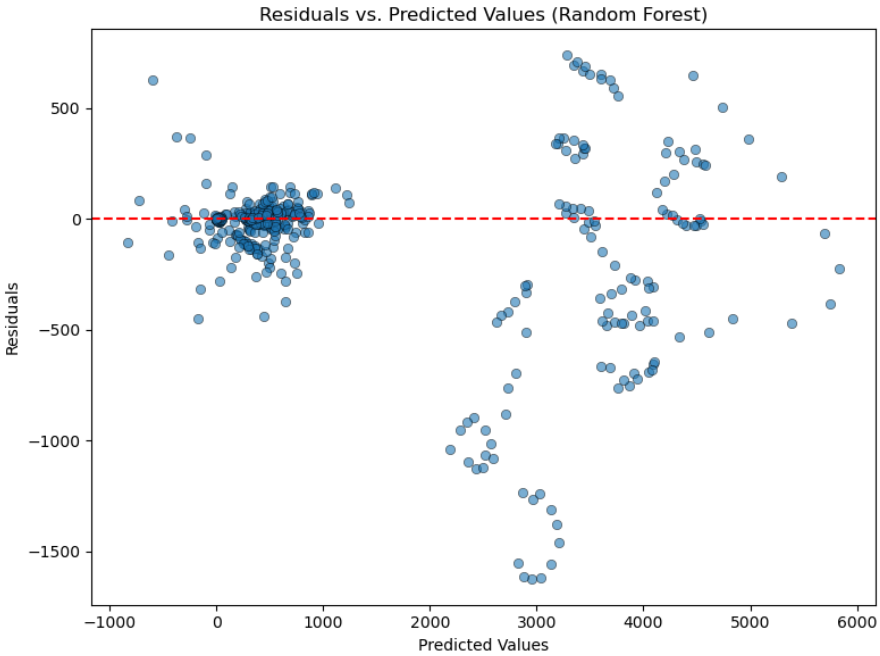


Figure A.5: Predicted values plotted against residuals, random forest

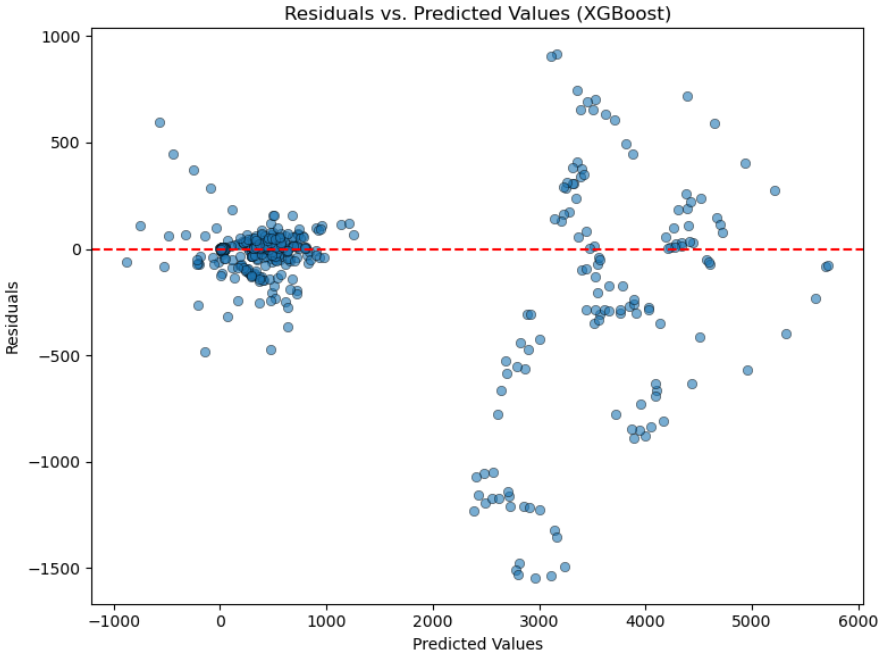


Figure A.6: Predicted values plotted against residuals, XGBoost

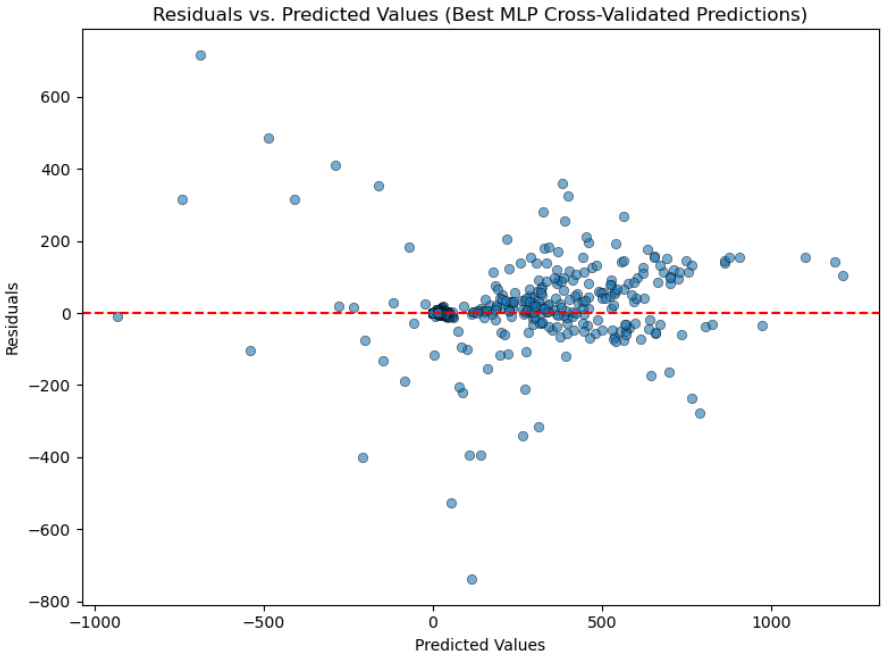


Figure A.7: Predicted values plotted against residuals, NN

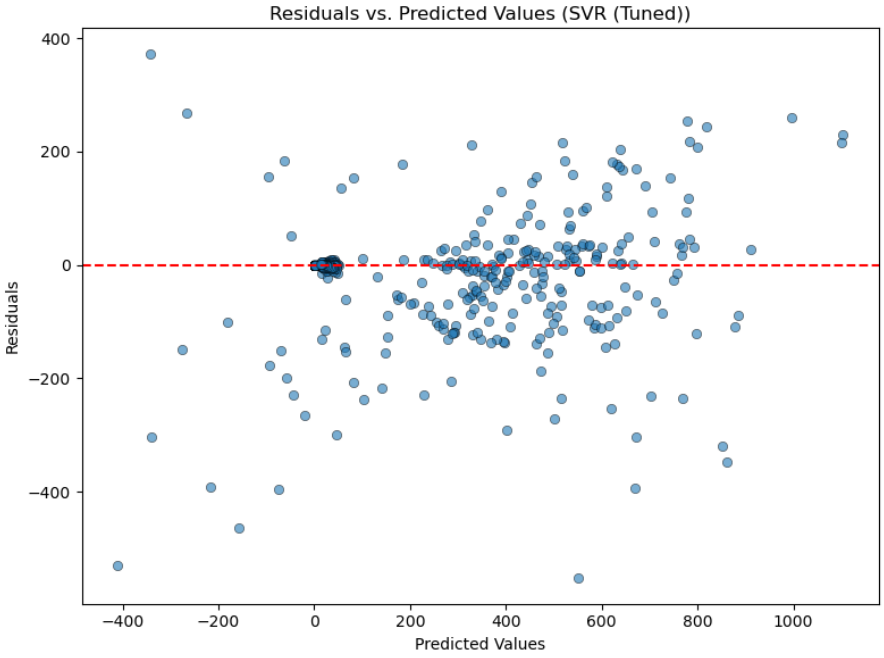


Figure A.8: Predicted values plotted against residuals, SVR

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY